AÑO II N.º 58

MICROHOBBY AMSTRAD Semanal

REVISTA INDEPENDIENTE PARA USUARIOS DE ORDENADORES AMSTRAD

AÑO II N.º 58

160 Ptas.
Canarias 165 pts.

EL MUNDO DEL PCW Y MUCHO MAS

ILIMITADOS TAMAÑOS DE LETRA PARA TUS PROGRAMAS

LAS PALABRAS EN LOGO

Peek y Poke: La chispa de tu Amstrad

Convierte el monitor de tu Amstrad en un televisor

¡Al asalto del castillo maldito!

HOBBY PRESS

TU PROGRAMA DE RADIO claro!

- Entrevistas a fondo
- Exitos en Soft
- Noticias en Hard
- Concursos

Prográmatelo: Sábados tarde de 5 a 7 horas.
En directo y con tu participación.

LA COPE A TOPE.
—RADIO POPULAR 54 EMISORAS O.M.—
En Barcelona Radio Miramar

COPE

Director Editorial
José I. Gómez-Centurión
Director Ejecutivo
Jasé M.ª Díaz
Redactor Jefe
Juan José Martínez
Diseño gráfico
Fernanda Chaumel
Colaboradores
Eduarda Ruiz
Javier Barcelá
David Sopuerta
Rabert Chatwin
Francisca Partalo
Pedra Sudán
Miguel Sepúlveda
Francisco Martín
Jesús Alonsa
Pedra S. Pérez
Amalio Gómez
Alberto Suñer
Secretaria Redacción
Carmen Santamaria
Fotografía
Carlas Candel
Chema Sacristán
Portada
Angel Luis Ganzález
Ilustradores
J. Igual, J. Pans, F. L. Frantán, J. Septien, Peja, J. J. Mara
Edita
HOBBY PRESS, S.A.
Presidente
María Andrina
Consejero Delegado
José I. Gómez-Centurián
Jefe de Producción
Carlas Peropadre
Marketing
Marta García
Jefe de Publicidad
Cancha Gutiérrez
Secretaria de Dirección
Pilar Arestizábal
Suscripciones
M.ª Rasa Ganzález
M.ª del Mar Calzada
Redacción, Administración y Publicidad
Ctra. de Irún km 12,400
(Fuencarral) 28049 Madrid
Pedidos y suscripciones:
734 65 00
Redacción: 734 70 12
Dto. Circulación
Paulina Blanco
Distribución
Caedis, S. A. Valencia, 245
Barcelana
Imprime
ROTEDIC, S. A. Crta. de Irún. Km. 12,450 (MADRID)
Fotocomposición
Novacomp, S.A.
Nicalás Morales, 38-40
Fotomecónica
GROF
Ezequiel Solana, 16
Depósito Legal:
M-28468-1985

MICROHOBBY
AMSTRAD
sumario

Año II • Número 58 • 21 al 27 de Octubre
160 ptas. (incluido I.V.A.)
Canarias, 155 ptas. + 10 ptas. sobretasa aérea
Ceuta y Melilla, 155 ptas.

STARSTRIKE II

Tu nueva misión es destruir los 22 planetas enemigos y lograr tener en tus manos el arma final.
Los mejores gráficos en 3D, por fin, en color.

FIREBIRD

EL MEJOR JUEGO TIPO ARCADE DEL AÑO

EL OBJETIVO ESTA CERCA ¿CONSEGUIRAS ALCANZARLO?

DISPONIBLE EN CASSETTE AMSTRAD AL PRECIO DE 2.700 pts.

PIDELO A SERMA, C/. CARDENAL BELLUGA, 21. 28028 MADRID Tels: 256 21 01/02 - 256 50 06/05/04

TITULO: _____
NOMBRE Y APELLIDOS: _____
DIRECCION: _____ CODIGO POSTAL: _____
PROVINCIA: _____ POBLACION: _____
FORMA DE PAGO: TALON BANCARIO ☐ CONTRA REEMBOLSO ☐ _____

SERMA

NUEVA AVENTURA DE DINAMIC

Siglo XXV. Los guerreros del planeta Kindos han atacado la Tierra, raptando a la princesa Doxaphin y robando todas las riquezas del Palacio Imperial.

El gran emperador Cophenix II Señor de los reinos Normax y Dinax, ha encomendado la misión de rescate a su fiel vasallo Mirdav, que recibirá como recompensa por su éxito la mano de la princesa y la mitad del fabuloso tesoro como dote.

Para conseguirlo tendrás que penetrar en el Castillo Kindos y enfrentarte con valentía a numerosos enemigos que intentarán hacer fracasar tu misión.

La clave del éxito reside en tu arma la joya más preciada del Imperio: la espada SGRIZAM, vencedora en mil batallas, forjadora de leyendas, inspiradora de pasiones.

Deja que su poder te guíe en esta fantástica aventura.

TUS ENEMIGOS

Atravesar los pasadizos del Castillo Kindos no es un paseo precisamente.

Te verás acechado por arañas gigantes, serpientes de colmillos letales, ratas enormes, patos zombies, esqueletos mutantes, afilados cuchillos voladores, etc. Todos los ingredientes para valorar mucho cuál será tu siguiente paso.

Cuando más seguro estés de haberte librado de una rata, probablemente un cuchillo atravesará limpiamente tu yugular.

Cuando más convencido estés de escapar al ataque de patos y esqueletos, te encontrarás ensartado por un sable enemigo.

Este Castillo, puedes tener la certeza, no es un hotel para el reposo.

NOVEDADES DE ERBE

Dos nuevos juegos, deliciosamente distintos, presenta ERBE soft como novedades: «**Las Tres Luces de Glaurung**» y **Desert Fox**. El primero consiste en una excelente aventura gráfica, en la que el protagonista se introduce, armado sólo con su espada, en l tenebroso castillo del Señor del Mal. Le acechan multitud de peligros, pero él debe conseguir las tres luces y salir del castillo. Estamos ante un excelente y entretenido juego de aventuras.

El segundo de ellos, **Desert Fox**, versa sobre la famosa batalla del desierto entre Rommel y Montgomery, en la Segunda Guerra Mundial. Esta vez, asumimos el papel de «Lobo Solitario», un inglés sobre el que descansa la ardua tarea de detener el aparentemente imparable avance de las tropas nazis. Por lo menos, la acción está asegurada.

OFERTA ESPECIAL I ANIVERSARIO

6 meses Gratis

de AMSTRAD CASSETTE

Suscríbete ahora a Microhobby Amstrad, o realiza tu renovación, y recibirás, totalmente gratis, un regalo de excepción: **una suscripción a Amstrad Cassette por seis meses.**

Cada cinta contiene los programas publicados por Microhobby Amstrad durante un mes.

Todos los programas de nuestras cintas se encuentran desprotegidos, con el objeto de facilitar su copia en disco y la revisión de los listados.

En cada cinta encontrarás:

— Apasionantes juegas llenas de acción y dinamismo.
— Utilidades con las que sacar mayor partida a tu ordenador.
— Rutinas en código máquina, para que las utilices en tus propios programas.
— Y pequeños trucos de programación, para que, poco a poca, te conviertas en un experto.

Recorta o copia el cupon que aparece cosido en las páginas de esta revista. **APROVECHA ESTA OFERTA UNICA, válida sólo para España hasta el 31 de noviembre de 1986.**

Vamos a ir ahora en sentido contrario. Puesto que la escritura en memoria ya no tiene ningún secreto para nosotros, vamos a intentar **«leer»** el contenido de una de las celdillas que la forman. Con PEEK lo conseguiremos.

PEEK es la sentencia complementaria de POKE. Si tecleamos:

POKE 20000,42

seguido de RETURN estamos almacenando en la dirección de memoria 20000 el número 42. Es decir, el ordenador coge el 42 y la guarda en la posición identificada como 20000. Y si ya teníamos algún valor almacenado en ese lugar, **¿qué ocurrirá?**

Todos los bytes tienen alguna información, aunque sea cero. Cuando intentamos cambiar esta información por otra *(42 en el ejemplo anterior)* simplemente ocurre eso: el nuevo valor se almacena y el antiguo se destruye.

Así que con:

POKE 20000,42

sustituimos el número que estuviera almacenado en la dirección 20000, borrándolo y anulándola, por el nuevo (42). Pero, **¿estamos seguros de que realmente ha cambiado el contenido de esa posición?** Veámoslo.

Teclee ahora:

PRINT PEEK (20000)

y, ¡vaya!, en la pantalla acaba de aparecer el 42 que anteriormente habíamos guardada en ese lugar.

Escriba a continuación:

POKE 20000,127

Teóricamente, al menos, en esta ocasión el byte 20000 de la memoria contendrá un número binario equivalente al decimal 127. **¿Lo comprobamos?**, ya sabemos hacerla, **¿no?** Simplemente escribiendo:

PRINT PEEK(20000)

vemos lo que ha ocurrida. Ya no se visualiza el número 42 como antes acurrió, sino que se ha cambiado por el 127, que es el último valor almacenado en la dirección 20000 por medio de una instrucción POKE.

Concretemos todo esto. La instrucción PEEK nos permite investigar dentro de la memoria y ver qué es lo que contiene. La forma general de esta función es:

PEEK*(dirección)*

y nos devuelve el valor del data almacenado en la posición de memoria especificada en **«dirección»**.

Este parámetro puede ser cualquier número entero comprendido entre 0 y 65535, **¿recuerda por qué?** Si no fuera así le recomendamos que eche un vistazo a nuestro artículo anterior y seguro que lo entenderá sin ningún problema. ¿Le suena de algo un «registro» de direcciones compuesta por 2 bytes-16 bits?

¿Qué ocurrirá si intentamos ver lo que hay en una posición de la memoria mayor de 65535? No se quede ahí parada. La mejor forma de comprender exactamente cómo funciona una instrucción y todas sus limitaciones es tomar los mandos de su **Amstrad** y comprobarlo sobre sus propias teclas. Pues escriba:

PRINT PEEK(70000)

por ejemplo. El valor de la dirección sobrepasa el límite que habíamos puesto, **¿no?**

Bueno, pues ahora en la pantalla no aparece un número que indique el contenido de esa posición, sino que el **Amstrad** la ha cambiado por un mensaje de error:

Overflow

Con él nos está indicando que se ha producido un «rebosamiento» de lo que llamamos **«registro de direcciones»**, o sea que hemos intentada llegar hasta una memoria que sobrepasa la capacidad de direccionamiento de nuestro ordenador *(65535)*.

La función PEEK nos devuelve un número que está comprendida entre 0 y 255. Tenga en cuenta que se trata del data almacenado en un byte *(8 bits)* y la mayor cantidad que puede representarse con 8 dígitos binarios es, precisamente, 255 tal cama vimos.

Por ser un valor numérica, podemos hacer con él todas las operaciones que el **Amstrad** permite realizar con números. Por ejemplo, sería válido asignar el contenido de una posición de memoria a una variable numérica:

contenido=PEEK*(20000)*

y a continuación:

PRINT contenida

también paner cama parámetra de PEEK el contenido de una variable. Las das instrucciones anteriores padrían convertirse en:

dirección=20000

seguido de:

contenida=PEEK*(dirección)*

y

PRINT contenida

donde **«dirección»** es, precisamente, la variable donde está guardado el valor de la dirección cuyo contenida queremos investigar.

Siempre padremas realizar cualquier operación matemática con el valor entregada por PEEK. Sería posible calcular el resultada de:

resultado=256*PEEK(dirección)
PRINT resultado

sin que nuestro **Amstrad** se sienta malesta y nas absequie, como tan frecuentemente hace, can un mensaje de error.

Esta facilidad es la que nos permitirá almacenar números mayores de 255 que era el límite, ¿na? Supongamos que queremos almacenar el número 256. Es evidente que siempre ocurrirá:

256 = 255 + 1

Primeros PASOS

Así que, resulto que 256 traducido a binario es un 1 seguido de ocho ceros, lo que implico que si un byte contiene 255 y le sumamos 1, en lugar de almacenar el 256 comenzaría a contar otra vez desde cero a la vez que dejaría por abandonada una unidad de noveno orden que no tendría cabida en él.

¿Qué hacer?

Pues tomar una sabia decisión: no abandonarla sino guardarla en la siguiente posición de la memoria. No estamos para desperdiciar nada. Lo que estamos haciendo es utilizar otra dirección para almacenar un valor que se irá incrementando en una unidad cada vez que la otra, la primitiva, comienza a contar de nuevo por cero.

Elijamos para este fin la 20000 y la 20001. Quedamos que la primera contendría el byte compuesto por los 8 ceros, luego:

```
POKE 20000,0
```

y la siguiente llevaría el 1 que nos ha sobrado, pues, ¡adelante!:

```
POKE 20001,1
```

guardaría un 1 indicando que tendríamos allí una unidad de noveno orden como antes dijimos. Y, **¿cuál es el valor decimal de estas unidades?** A calcular tocan.

```
2^8 = 256
```

Lo mismo ocurrirá con el resto de cifras de este byte *(el 20001)*: que su valor relativo irá multiplicado por 256.

Para conseguir leer correctamente la información que hay almacenada en estas dos direcciones tendremos que operar de la siguiente manera:

```
dato=PEEK(20000)+256*PEEK(20001)
```

para después imprimirlo con:

```
PRINT dato
```

e inevitablemente se visualizará el 256 que andábamos buscando.

Con todas estas premisas podríamos seguir contando en la dirección 20000 partiendo de cero como si no hubiera pasado nada y utilizar la siguiente, la 20001 para anotar el número de veces que la primera alcanzó el valor 255.

Repetimos. La primera dirección es una especie de contador donde se van anotando números comprendidos entre 0 y 255 y se le llama **«Byte menos significativo».** La segunda lo que hace es como si fuera contar de 256 en 256, ya que se incrementa cuando la primera sobrepasa 255. Se la conoce como **«Byte más significativo»**.

Bueno, **¿y para qué vale todo esto?** ¡Ah!, ¿no ha quedado claro cuál era el fin de nuestro trabajo? Es bastante evidente que lo que pretendíamos era intentar almacenar directamente en la memoria números mayores de 255 y modestamente creemos que siguiendo este método lo vamos a conseguir.

Para guardarlo, partiremos el número para poder colocarlo en dos direcciones de memoria. Una tendría el número de veces que contiene 256: aquí tenemos una pista. Si hacemos la división entera del número entre 256 ya sabemos el valor que habrá que almacenar en el byte más significativo.

Elijamos el número, **¿le parece?** Las direcciones serán las ya conocidas 20000 y 20001. ¿Se acuerda del valor máximo que se puede almacenar en 16 bits *(ó 2 bytes como en este caso)*? La respuesta es 65535. Sin problemas, **¿no?** Pues tomemos uno al azar: 23456.

```
número=23456
```

Calculamos las veces que contiene 256 con:

```
más=23456/256
```

y lo almacenamos en la dirección del byte más significativo mediante:

```
POKE 20001,más
```

Nos quedo ahora el otro, el «menos significativo», pero tan importante como el anterior. Contendrá el resto de dividir el número entre 256. **¿Lo calculamos?**

```
menos=número MOD 256
```

y a continuación lo almacenamos, ¿no?

```
POKE 20000,menos
```

Estará de acuerdo con nosotros que en este momento tenemos guardado el número en dos posiciones de memoria, la 20000 y la 20001 y que, por tanto, hemos conseguido nuestro objetivo: almacenar números mayores de 255.

De manera que **«es menester»** generalizar este proceso a cualquier cantidad y a cualquier dirección. Para guardarlo en la memoria daremos los siguientes pasos:

a) Calcular la cantidad de veces que el número contiene 256 y almacenarlo en el byte más significativo.

```
POKE dirección+1, número/256
```

b) Introducir el resto de la división entre el número y 256 en el byte menos significativo.

```
POKE dirección, número MOD 256
```

Y todo resuelto, ya lo tenemos a buen recaudo.

¿Cómo realizar el proceso contrario? Ahora queremos sacar la información de dos bytes. Resumamos el proceso:

a) Tomar el contenido del byte más significativo y multiplicarlo por 256.

b) Sumar el resultado obtenido al valor que está almacenado en el menos significativo y ya tendremos en nuestras manos el número que necesitamos. O lo que es lo mismo:

Pero calculemos este número en binario. ¿Se atreve a hacerla suma?

```
255 = 11111111
  1 = 00000001   +
-----------------
256 = 100000000
```

número = 256 * PEEK(dirección + 1) + PEEK(**dirección**)

Recuerde que en dos bytes sólo se pueden guardar números comprendidos entre 0 y 65535 así que no intente repetir este proceso para números mayores yo que se llevará alguna que otro desagradable sorpresa.

Y como colofón a todas estas parrafadas vamos a efectuar un breve análisis a un programa que pone en práctica casi todo lo que hemos visto a lo largo del artículo. También nos hemos auxiliado un poco del antiguo Programa I de nuestra anterior reunión, pero solamente para seguir su misma forma y estructura.

¿Qué conseguimos con él? Aparentemente nada, sólo vamos a **«andar»** por la memoria tal como dice su título. Lo exploraremos en todas sus direcciones y a lo mejor hasta nos atreveremos a modificar alguno de sus **«bytes»** a ver qué pasa. En caso de que la ejecución del programa se detenga peligrosamente, recuerde, apague el ordenador y comience de nuevo: no se le ha estropeado.

Programa uno

Su estructura es muy sencilla. Consta de un programa principal, que será el que dirija el **«cotarro»**, y de una serie de rutinas cuya misión es realizar unas tareas muy concretitas y sencillas.

Vamos con el programa principal. Comienza en la línea 10 y lo primero que hace es llamar a una rutina que inicializará la pantalla, las distintas ventanas de texto y todo lo que sea necesario. ¡Ah!, su primera línea es la 5000.

A continuación vamos a otra rutina **(GOSUB 6000)** que se encargará de visualizarnos el menú de las opciones de que disponemos. Pero además nos permitirá elegir una de ellas para que el programa **«lea»** o **«escriba»** en cualquier posición de la memoria del **Amstrad.**

Después de una serie de instrucciones, que imprimen la cabecera de un informe, tras habernos decidido por una de las opciones *(líneas 40 a 60)*, nos metemos en un bucle sin fin que nos permitirá manejar todas las direcciones de memoria que deseemos sin tener que comenzar de nuevo la ejecución del programa. Es el limitado entre las líneas 70 y 130.

Y, **¿qué es lo que se hace en su interior?** Sigámosle paso a paso. En la línea 80 se nos pide la dirección a tratar y el valor que nosotros tecleamos lo almacena en la variable **«dirección».**

Si no es correcto, pone un aviso de error y le da el tipo número 34. Esta es una de las características más importantes de este programa: nosotros mismos gestionamos como un error los datos que no están de acuerdo con las características especificadas *(en este caso está comprendida entre 0 y 65535)*.

Con el EN...GOSUB de la 100 desviamos la ejecución del programa hacia la rutina de **«lectura»** o **«escritura»,** según la opción elegida.

```
10 REM PROGRAMA 1
20 GOSUB 5000:REM INICIALIZAR
30 GOSUB 6000:REM MENU
40 CLS
50 PRINT"DIRECCION","CONTENIDO"
60 PRINT"---------","---------"
70 WHILE -1
80 INPUT #2,"DIRECCION  (99999-FIN):
 ",direccion
90 IF direccion<0 OR direccion>65535
THEN canal=2:ERROR 34
100 ON opcion GOSUB 1000,2000
110 PRINT #1,direccion,contenido
120 CLS #2:CLS #3
130 WEND
1000 REM LECTURA
1010 contenido=PEEK(direccion)
1020 RETURN
2000 REM ESCRITURA
2010 INPUT #3,"CONTENIDO: ",contenido
2020 IF contenido<0 OR contenido>255
THEN canal=3:ERROR 35
2030 POKE direccion,contenido
2040 RETURN
3000 REM RUTINA TRATAMIENTO DE ERROR
3010 IF direccion=99999 THEN RESUME 3
0
3020 CLS #canal
3030 IF ERR=33 THEN RESUME 30
3040 IF ERR=34 THEN PRINT #2,"DIRECCI
ON INCORRECTA":GOSUB 4000:RESUME 80
3050 IF ERR=35 THEN PRINT #3,"DATO IN
CORRECTO":GOSUB 4000:RESUME 2010
4000 REM RUTINA DE RETARDO
4010 FOR i=0 TO 1000:NEXT
4020 CLS #canal
4030 RETURN
5000 REM INICIALIZACION
5010 ON ERROR GOTO 3000
5020 WINDOW #1,1,40,4,22
5030 WINDOW #2,1,40,24,24
5040 WINDOW #3,1,40,25,25
5050 RETURN
6000 REM MENU
6010 CLS:CLS #1:CLS #2:CLS #3
6020 LOCATE 8,6
6030 PRINT"ANDADURA POR LA MEMORIA"
6040 LOCATE 8,7
6050 PRINT"************************"
6060 LOCATE 13,9
6070 PRINT"1.- LEER"
6080 LOCATE 13,11
6090 PRINT"2.- ESCRIBIR"
6100 LOCATE 12,14
6110 INPUT"ELIGE OPCION: ",opcion
6120 IF opcion<1 OR opcion>2 THEN ERR
OR 33
6130 RETURN
```

Ya lo único que nos queda para terminar el bucle es visualizar la dirección y el contenido de la misma *(línea 110)* y limpiar el resto de las ventanas definidas *(120)*.

Puede ir usted mismo analizando todas las rutinas que emplea este programa. Para nosotros, ya lo hemos dicho, lo más importante es el tratamiento que da a los errores cometidos al teclear los datos.

Por ejemplo, en el rutina que genera el menú de opciones, **¿entre qué valores podemos escoger?** Dos opciones, dos valores: 1 y 2. **¿De acuerdo?** Por eso en la línea 6120 decimos que se ha producido el error 33 en el caso que la variable **«opción»** no contenga uno de estos valores.

Y algo semejante ocurre en la 2010, donde determinamos que se da el error 35 cuando el valor que pretendemos introducir en un byte está fuera de los márgenes permitidos **(0-255).**

Cuando se produzca un error, **¿hacia donde se desviará la ejecución del programa para tratarlo convenientemente?**

Observe la línea 5010:

5010 ON ERROR GOTO 3000

Bueno, pues nuestra pregunta ha quedado ya contestada: hacia la rutina de tratamiento de error que comienza en la línea 3000.

Ya estamos colocados ahí. **¿Ahora qué?** Primero investigaremos si lo que ocurre es verdaderamente un error o es que estamos indicando al **Amstrad** que determine el proceso que tiene entre manos. Si es así, nos vuelve a presentar el menú para continuar nuestro trabajo. Recuerde que 99999 es el valor de **«dirección»** elegido para tal fin.

En caso contrario, analiza el tipo de error. Si es el 33 entiende que nos hemos equivocado en la **«opción»** y nos devuelve otra vez al menú para que lo hagamos correctamente.

Cuando sea el 34 decidimos que la «dirección» que pretendíamos tratar no era correcta. Bueno, nos informa del hecho y nos da una nueva oportunidad reanudando la ejecución del programa en la línea 80 *(RESUME 80)*.

Sólo nos queda ver qué ocurre cuando el tipo de error sea 35. Sencillo, el dato que queremos almacenar en un byte no está comprendido entre 0 y 255. El **Amstrad** nos avisa y devuelve el control de la línea 2010 donde se nos pedirá un nuevo valor.

Sencillo, ¿verdad? Lo único que se necesita es un poco de práctico para ver claramente la manera de tratar este tipo de errores de un modo simple, pero a la vez atando todos los cabos.

Y nada más. Recuerde:

POKE = HURGAR
PEEK = HUSMEAR

En sus manos los dejamos.

ANALISIS

DESPUES DE TANTO TIEMPO...

El Basic Locomotive es multitarea, es decir, permite realizar más de una subrutina «a un tiempo». Realmente, esto no es así; la ejecución es secuencial, sólo que, tan rápida, que a nosotros nos parecerá simultánea.

por José María Diaz

Hay dos formas de usar esta sorprendente facilidad del Basic **Amstrad**: la sentencia EVERY y la sentencia AFTER.

Esta última permite que, después de transcurrido determinado tiempo, el Basic bifurque a determinada rutina, la ejecute, y **«retorne»** a la que estaba realizando inicialmente. Como el **Amstrad** posee 4 temporizadores, podemos **«liar»** a 4 rutinas más el programa principal de esta forma.

El programa de esta semana muestra tan sólo las posibilidades que se esconden tras la sentencia AFTER.

Sugerencias para aprovecharla se nos ocurren, así, a bote pronto, ver qué ha pasado con la puntuación de un juego después de un tiempo, o comprobar la pulsación de cierta tecla. Seguro que vuestra imaginación os sugerirá varios cientos de aplicaciones más.

En fin, vamos a estudiar el programa con detalle.

10—20 La usual **«sarta»** de REMs indicando de qué va el programa.

30 Elección del modo de pantalla: texto más claro y un CLS gratis cada vez que se ejecuta.

40 Inicialización de las variables **«sum, rutina y flag».** El **Amstrad** las pondría a cero él solito, pero es una buena práctica de programación indicar explícitamente las cosas.

50—80 Orden a la máquina de preparar las 4 temporizadores *(0 al 3)* y ejecutar las rutinas 1—4 una vez transcurrido un tiempo, que se indica en unidades de 1/50 segundos. Así, en la línea 50 se le dice al micro que salte que salte a la rutina de la línea190 después de que hayan pasado 5 segundos *(250 unidades de 1/50)*, y se asigna todo esto al temporizador cero.

90—100 El usuario se entera del asunto. Mientras las 4 rutinas aguardan para ejecutarse, el programa principal sigue funcionando.

120—140 El **«programa principal».** En este caso, se limita a incrementar el valor de la variable «sum», hasta que el valor de **«flag»** cambie, cosa que ocurrirá cuando se ejecute la rutina 4.

160—170 Fin. El programa principal sólo llega aquí cuando **«flag»** vale 1.

180 Detiene el programa. Pruebe a ver qué pasaría si se quita; merece la pena.

190—230 Primera subrutina, llamada cuando han transcurrido 5 segundos PARA EL TEMPORIZADOR CERO. En este momento, el bucle WHILE...WEND de las líneas 120—140 se **«duerme».**

240—390 Lo mismo para las otras tres rutinas.

380 Flag se pone a uno. Condición de salida para el programa principal *(líneas 120—140)*.

```
10 REM DESPUES DE TANTO TIEMPO
...
20 REM AMSTRAD SEMANAL
30 MODE 2
40 sum=0:rutina=0:flag=0
50 AFTER 250,0 GOSUB 190
60 AFTER 500,1 GOSUB 240
70 AFTER 750,2 GOSUB 290
80 AFTER 1000,3 GOSUB 340
90 PRINT"Los cuatro AFTERs est
an definidos, listos para llam
ar a cada subrutina DESPUES d
el tiempo transcurrido"
100 PRINT"Al comenzar, suma es
 ";sum;" mientras que rutina e
s ";rutina
110 PRINT
120 WHILE flag <>1
130 sum=sum+1
140 WEND
150 PRINT
160 PRINT"Las cuatro subrutina
s han sido ejecutadas"
170 PRINT"Suma es ahora ";sum
180 END
190 REM PRIMER AFTER
200 PRINT" PRIMER AFTER"
210 rutina=rutina+1
220 PRINT"Suma es ahora ";sum;
" mientras que rutina es ";rut
ina
230 RETURN
240 REM SEGUNDO AFTER
250 PRINT" SEGUNDO AFTER"
260 rutina=rutina+1
270 PRINT"Suma es ahora ";sum;
" mientras que rutina es ";rut
ina
280 RETURN
290 REM TERCER AFTER
300 PRINT" TERCER AFTER"
310 rutina=rutina+1
320 PRINT"Suma es ahora ";sum;
" mientras que rutina es ";rut
ina
330 RETURN
340 REM CUARTO AFTER
350 PRINT" CUARTO AFTER"
360 rutina=rutina+1
370 PRINT"Suma es ahora ";sum;
" mientras que rutina es ";rut
ina
380 flag=1
390 RETURN
```

NUMEROS ALEATORIOS (y II)

En este capítulo vamos a tratar de una función que resulta vital a la hora de confeccionar cualquier tipo de juego, pero además para aquellos que no les interesen los juegos, hemos aprovechado este tema para hacer una pequeña introducción a la inteligencia artificial desde lenguaje máquina.

La función en sí se trata de aquello que es capaz de generar números aleatorios.

Desde Basic para la generación de números aleatorios, disponemos del siguiente comando:

RND

que puede utilizarse o bien solo o acompañado de:

RANDOMIZE

Ahora bien, cuando nos movemos al nivel del lenguaje máquina, no disponemos de ninguna llamada al firmware que nos proporcione este tipo de números, por lo que debemos crear una rutina que sea capaz de producir dicho números.

Por supuesto, nuestra rutina nunca será tan eficaz como la que posee el sistema, aunque debemos decir que esta última tampoco es perfecta, ya que los números aleatorios que genera no lo son realmente, pero se obtiene una buena aproximación.

Para la confección de nuestra rutina nos ayudaremos del reloj interno que contiene el ordenador.

Dado que no conocemos los datos que contiene el contador interno, deberemos llamar a una rutina del firmware que los proporcione, ésta es la siguiente:

PROPORCIONA EL TIEMPO TRANSCURRIDO. #BD0D

Nos da el tiempo transcurrido desde que se ha enchufado el ordenador.

Condiciones de entrada.

No se precisan.

Condiciones de salida.

Los registros DE y HL contienen el valor del contador *(D contiene el byte más significativo y L el menos significativo)*. Se preservan los demás registros.

El contador del reloj se incrementa cada 1/300 segundos, esto se produce en cada interrupción que provoca el sistema. Así pues, tomando los dos bytes menos significativos del contador del reloj, construimos nuestra rutina de números aleatorios.

En nuestro caso deseamos obtener números aleatorios entre 0 y 128, por lo que deberemos delimitar el valor que nos proporcione dicho contador.

Para ello podemos proceder de la siguiente forma:

```
CALL #BD0D
LD A,L
XOR H
AND 127
INC A
```

De esta forma, en el acumulador estaría almacenado el número aleatorio obtenido. En este caso hemos realizado una operación «XOR» entre los bytes menos significativos del contador, pero podríamos haber utilizado cualquier otra como **«OR»** o **«AND»**.

Vamos a centrarnos ahora en el programa que hemos preparado en Basic y que hemos traducido a código máquina.

Se trata de un juego, en el cual el ordenador intentará adivinar un número que nosotros hayamos pensado previamente.

El proceso del juego sería el siguiente:

1. Pensamos un número.
2. El ordenador nos da un resultado:
   — Si el resultado es el número pensado por nosotros, se termina el juego.
   — Si el resultado es distinto, deberemos indicarle con la tecla de cursor, si nuestro número es mayor o menor.
3. Se calcula un nuevo número y retorna a la fase 2.

Para indicar al ordenador si nuestro número es mayor o menor al que se ha pensado, utilizaremos las teclas del cursor de la siguiente forma:

Cursor arriba: para indicar que el número que hemos pensado es mayor.

Cursor abajo: para indicar que el número que se ha pensado es menor.

En realidad un juego de este tipo si únicamente estuviera basado en la generación de números aleatorios, no tendría ningún aliciente, ya que el ordenador podría adivinar nuestro número en la primera jugada o bien después de muchísimos intentos.

Ahí es donde radica la diferencia entre la pura aleatoriedad y la inteligencia artificial o la capacidad de «pensar» de nuestro ordenador a través de una serie de algoritmos que hayamos introducido en nuestro programa.

Así pues, nuestro programa está dotado de un pequeño **«cerebro»** basado en dos algoritmos diferentes, con el cual será capaz de adivinar el número que hayamos pensado en un número no superior a ocho jugadas.

Para ello nuestra rutina toma un primer número al azar, que es proporcionado por la rutina aleatoria anteriormente citada.

Seguidamente coloca los topes superior e inferior en dos variables:

SUPREM...128 INFIM...0

A partir de este momento el programa actuará de forma diferente según le indiquemos que el número pensado a superior o inferior al que nos ha dado en último término.

Si el número indicado por el ordenador es mayor al número a adivinar, entonces efectuará el siguiente proceso:

— Actualizará la variable «SUPREM» colocando en ella el número actual.

— Calculará la diferencia entre los valores indicados por **«INFIM»** y dicho número.

— Dividirá dicha diferencia por dos.

— Y por último se restará ese valor al último número, obteniendo de esta forma el nuevo valor.

Escrito en lenguaje máquina, nos quedaría de la siguiente forma:

```
INFER: LD B,A
LD,A(NUMER)
LD (SUPREM),A; Repone el valor superior
SUB B; Dif. entre NUMER e INFIM
SRL A; Divide la dif. por dos
LD A, (NUMER)
SUB B; Resta la dif. al número ant.
LD (NUMER),A; Se obtiene un nuevo valor
RET
```

Cuando el número indicado por el ordenador sea el menor al que se

```
10 MODE 1
20 LOCATE 5,10:PRINT "PIENSA UN NUM
ERO Y PULSA UNA TECLA"
30 GOSUB 100
40 MODE 1
50 N=INT (RND*127)+1:INFIM=0:SUPREM
=128
60 GOSUB 120
70 IF INKEY(0)=0 THEN 200
80 IF INKEY(2)=0 THEN 220
90 GOTO 70
100 WHILE INKEY$="":WEND
110 RETURN
120 LOCATE 10,10:PRINT "EL NUMERO E
S: ";N
130 LOCATE 5,13:PRINT "          (
S/N)              "
140 IF INKEY (60)=0 THEN END
150 IF INKEY (46)=0 THEN 170
160 GOTO 140
170 LOCATE 5,13:PRINT "TU NUMERO ES
 MAYOR ";CHR$(240);" O MENOR ";CHR$
(241)
180 RETURN
190 RETURN
200 INFIM=N:N=N+INT((SUPREM-N)/2)
210 GOTO 60
220 SUPREM=N:N=N-INT((N-INFIM)/2)
230 GOTO 60
```

Código MAQUINA

haya pensado, se procederá como sigue:

— Se actualizará la variable 'INFIM' colocando en ella el último valor.

— Se calculará la diferencia entre los valores indicados por 'SUPREM' y el número actual.

— Dividirá dicha diferencia por dos.

— Y finalmente se sumará al valor actual, obteniendo así el nuevo número.

Así se escribiría dicho procedimiento en lenguaje ensamblador:

```
SUPER: LD A,(NUMBER)
LD (INFIM), A; Acualiza el valor inferior
LD B,A
LD A,(SUPREM)
SUB B; Dif. entre SUPREM y NUMBER
SRL A; Divide la dif. por dos
LD B,A
LD A,(NUMBER)
ADD A,B; Suma la dif. al número actual
LD (NUMBER),A; Se obtiene el nuevo valor
RET
```

Así pues, mediante estos dos algoritmos, el programa será capaz de adivinar cualquier número que hayamos pensado en el intervalo 1-127, en un máximo de ocho intentos.

```
10          ORG   #A000
20          LD    A,1
30          CALL  #BC0E
40          LD    HL,#050A
50          CALL  #BB75
60          LD    HL,TEXT3
70          CALL  PRINT
80          CALL  #BB18
90          CALL  #BB18
100         LD    A,1
110         CALL  #BC0E
120         XOR   A
130         LD    (INFIM),A
140         LD    A,128
150         LD    (SUPREM),A
160 RND:    CALL  #BD0D
170         LD    A,L
180         XOR   H
190         AND   127
200         INC   A
210         LD    (NUMER),A
220 TECLA:  CALL  PINUM
230         JR    CHECK
240 VUELV:  LD    A,66
250         CALL  #BB1E
260         RET   NZ
270 NOTE1:  XOR   A
280         CALL  #BB1E
290         JR    Z,NOTE
300         CALL  SUPER
310         JR    TECLA
320 NOTE:   LD    A,2
330         CALL  #BB1E
340         JR    Z,NOTE1
350         CALL  INFER
360         JR    TECLA
370
380 SUPER:  LD    A,(NUMER)
390         LD    (INFIM),A
400         LD    B,A
410         LD    A,(SUPREM)
420         SUB   B
430         SRL   A
440         LD    B,A
450         LD    A,(NUMER)
460         ADD   A,B
470         LD    (NUMER),A
480         RET
490 INFER:  LD    A,(INFIM)
500         LD    B,A
510         LD    A,(NUMER)
520         LD    (SUPREM),A
530         SUB   B
540         SRL   A
550         LD    B,A
560         LD    A,(NUMER)
570         SUB   B
580         LD    (NUMER),A
590         RET
600 CHECK:  LD    HL,#050D
610         CALL  #BB75
620         LD    HL,TEXT2
630         CALL  PRINT
640 BUCCH:  LD    A,60
650         CALL  #BB1E
660         RET   NZ
670         LD    A,46
680         CALL  #BB1E
690         JR    Z,BUCCH
700         LD    HL,#050D
710         CALL  #BB75
720         LD    HL,TEXT4
730         CALL  PRINT
740         CALL  #BB18
750         CALL  #BB18
760         JR    VUELV
770 ;
780 ; IMPRIME NUMEROS
790 ; DECIMALES
800 ;
810 ;
820 PINUM:  LD    HL,#0A0A
830         CALL  #BB75
840         LD    HL,TEXT1
850         CALL  PRINT
860         LD    A,(NUMER)
870         SCF
880         LD    H,0
890         LD    L,A
900         INC   HL
910         LD    A,47
920         LD    DE,100
930 CIEN:   INC   A
940         SBC   HL,DE
950         JR    NC,CIEN
960         CALL  PINTN
970         LD    DE,10
980 DIEZ:   INC   A
990         SBC   HL,DE
1000        JR    NC,DIEZ
1010        CALL  PINTN
1020        ADD   A,L
1030        CALL  PINTN
1040        RET
1050 PINTN: CALL  #BB5A
1060        LD    A,47
1070        JR    NZ,PAS
1080        INC   HL
1090 PAS:   ADD   HL,DE
1100        INC   HL
1110        RET
1120 PRINT: LD    A,(HL)
1130        CP    255
1140        RET   Z
1150        CALL  #BB5A
1160        INC   HL
1170        JR    PRINT
1180 TEXT1: DEFM "EL NUMERO ES: "
1190        DEFB 255
1200 TEXT2: DEFM "          (S/N)          "
1210        DEFB 255
1220 TEXT3: DEFM "PIENSA UN NUMERO Y PUSA UNA TECLA"
1230        DEFB 255
1240 TEXT4: DEFM "TU NUMERO ES MAYOR "
1250        DEFB 240
1260        DEFM " O MENOR "
1270        DEFB 241,255
1280        DEFB 255
1290 NUMER: DEFS 1
1300 SUPREM: DEFB 128
1310 INFIM: DEFB 0
```

```
BUCCH  A08A  CHECK  A07E  CIEN   A0C4
DIEZ   A0CF  INFER  A068  INFIM  A163
NOTE   A046  NOTE1  A03B  NUMER  A161
PAS    A0E4  PINTN  A0DC  PINUM  A0AB
PRINT  A0E7  RND    A025  SUPER  A052
SUPREM A162  TECLA  A030  TEXT1  A0F1
TEXT2  A100  TEXT3  A11F  TEXT4  A141
VUELV  A035
```

```
10 REM *PROGRAMA CARGADOR*
20 FOR N=&A000 TO &A164
30 READ A:SUMA=SUMA+A
40 POKE N,A
50 NEXT
60 IF SUMA<>36480 THEN PRINT "ERROR
 EN DATAS"
70 DATA 62,1,205,14,188,33,10
80 DATA 5,205,117,187,33,31,161
90 DATA 205,231,160,205,24,187,205
100 DATA 24,187,62,1,205,14,188
110 DATA 175,50,99,161,62,128,50
120 DATA 98,161,205,13,189,125,172
130 DATA 230,127,60,50,97,161,205
140 DATA 171,160,24,73,62,66,205
150 DATA 30,187,192,175,205,30,187
160 DATA 40,5,205,82,160,24,234
170 DATA 62,2,205,30,187,40,238
180 DATA 205,104,160,24,222,58,97
190 DATA 161,50,99,161,71,58,98
200 DATA 161,144,203,63,71,58,97
210 DATA 161,128,50,97,161,201,58
220 DATA 99,161,71,58,97,161,50
230 DATA 98,161,144,203,63,71,58
240 DATA 97,161,144,50,97,161,201
250 DATA 33,13,5,205,117,187,33
260 DATA 0,161,205,231,160,62,60
270 DATA 205,30,187,192,62,46,205
280 DATA 30,187,40,243,33,13,5
290 DATA 205,117,187,33,65,161,205
300 DATA 231,160,205,24,187,205,24
310 DATA 187,24,138,33,10,10,205
320 DATA 117,187,33,241,160,205,231
330 DATA 160,58,97,161,55,38,0
340 DATA 111,35,62,47,17,100,0
350 DATA 60,237,82,48,251,205,220
360 DATA 160,17,10,0,60,237,82
370 DATA 48,251,205,220,160,133,205
380 DATA 220,160,201,205,90,187,62
390 DATA 47,32,1,35,25,35,201
400 DATA 126,254,255,200,205,90,187
410 DATA 35,24,246,69,76,32,78
420 DATA 85,77,69,82,79,32,69
430 DATA 83,58,32,255,32,32,32
440 DATA 32,32,32,32,32,32,32
450 DATA 40,83,47,78,41,32,32
460 DATA 32,32,32,32,32,32,32
470 DATA 32,32,32,32,32,32,255
480 DATA 80,73,69,78,83,65,32
490 DATA 85,78,32,78,85,77,69
500 DATA 82,79,32,89,32,80,85
510 DATA 83,65,32,85,78,65,32
520 DATA 84,69,67,76,65,255,84
530 DATA 85,32,78,85,77,69,82
540 DATA 79,32,69,83,32,77,65
550 DATA 89,79,82,32,240,32,79
560 DATA 32,77,69,78,79,82,32
570 DATA 241,255,255,0,128,0,0
```

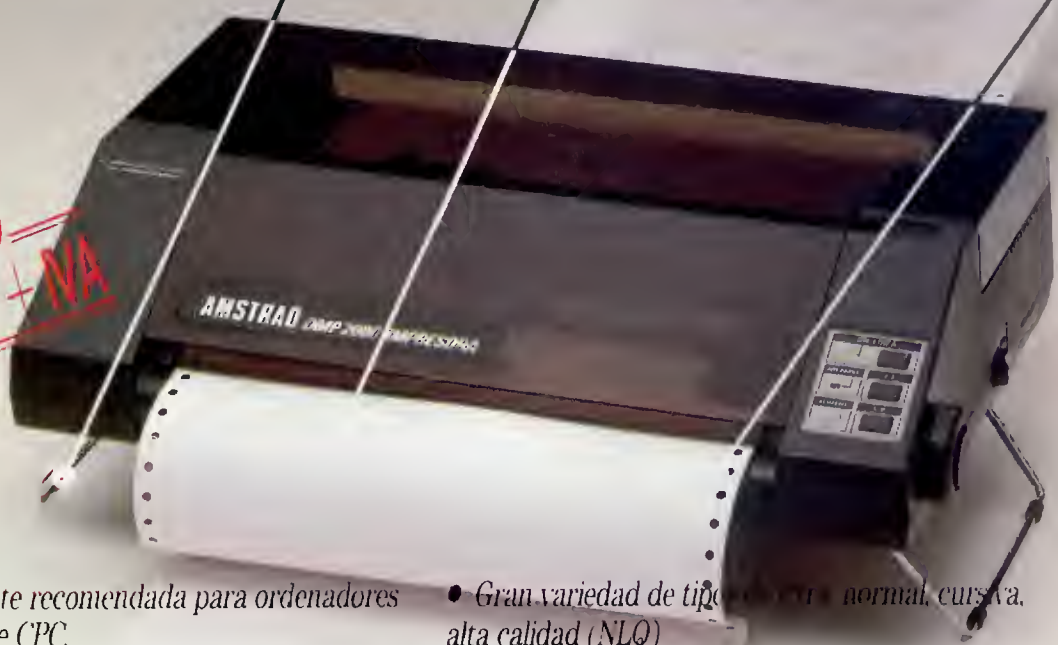
AMSTRAD DMP 2000

NO ENCONTRARA UNA IMPRESORA QUE LE HAGA TAN BUEN PAPEL.

Cómodo sistema de carga frontal del papel.

Admite diferentes anchos de papel, tanto continuo (de 114 a 254 mm.) como hojas sueltas (102 a 241 mm.)

Soportes abatibles que permiten colocar el papel bajo la impresora.



POR SOLO 39.500 PTAS + IVA

- *Especialmente recomendada para ordenadores AMSTRAD serie CPC.*
- *Conectable a cualquier ordenador con interface centronics.*
- *Velocidad de impresión de 105 caracteres por segundo.*
- *Gran variedad de tipos de letra: normal, cursiva, alta calidad (NLQ)*
- *40, 66, 80 y 132 caracteres por columna.*
- *Impresión de gráficos punto a punto en diferentes densidades.*
- *96 caracteres ASCII y 8 subjuegos internacionales.*

¡¡Increible!!

GRUPO INDESCOMP AMSTRAD ESPAÑA

C/. Aravaca, 22. 28040 Madrid. Tel. 459 30 01. Télex 47660 INSC E. Fax 459 22 92. Delegación en Cataluña: C/. Tarragona, 110. Tel. 325 10 58. 08015 Barcelona.

AMSCASTILLO

Los juegos de aventuras nos hacen vivir las más grandes epopeyas. Castillos, fantasmas, laberintos, princesas en peligro y todo tipo de seres imaginarios nos amenazan y asaltan desde la pantalla. Grupo de Asalto es un ejemplo de todo esto, de lo más vivo.

por Daniel Calvo y Fernando García

Sólo 664-6128

PRESENTACION: 7
GRAFICOS: 5
ANIMACION: 5
ADICCION: 5

Después de una pantalla de presentación, en la que se te preguntará si quieres instrucciones, el programa te pedirá el nombre de los dos jugadores, pues este programa necesita el concurso de dos personas. Pasado este requisito de los nombres, os encontraréis en la primera parte de las dos que consta este programa, que es el laberinto.

Laberinto: empezará jugando el primer jugador, aunque de todas formas en la parte baja de la pantalla, se expondrá el nombre del jugador y las vidas que tiene. El laberinto tardará un poco en aparecer, pero no es ningún problema. En el laberinto, tu propósito es el de intentar pasarlo con el mayor número posible de vida y en el menor tiempo que puedas, pues si el contador llega a cero perderás una vida. A la vez, si te chocas con las paredes o con tu propio rastro también perderás una vida. También verás que por el camino se encuentran unos corazones que te darán puntos.

Una vez que los dos paséis este trámite, entraréis en el castillo. Es importantísimo el número de puntos que consigas, pues de éste depende el número de disparos que luego obtendrás en el castillo, aunque puedes pasar sin vidas, pero tu número de disparos será muy inferior.

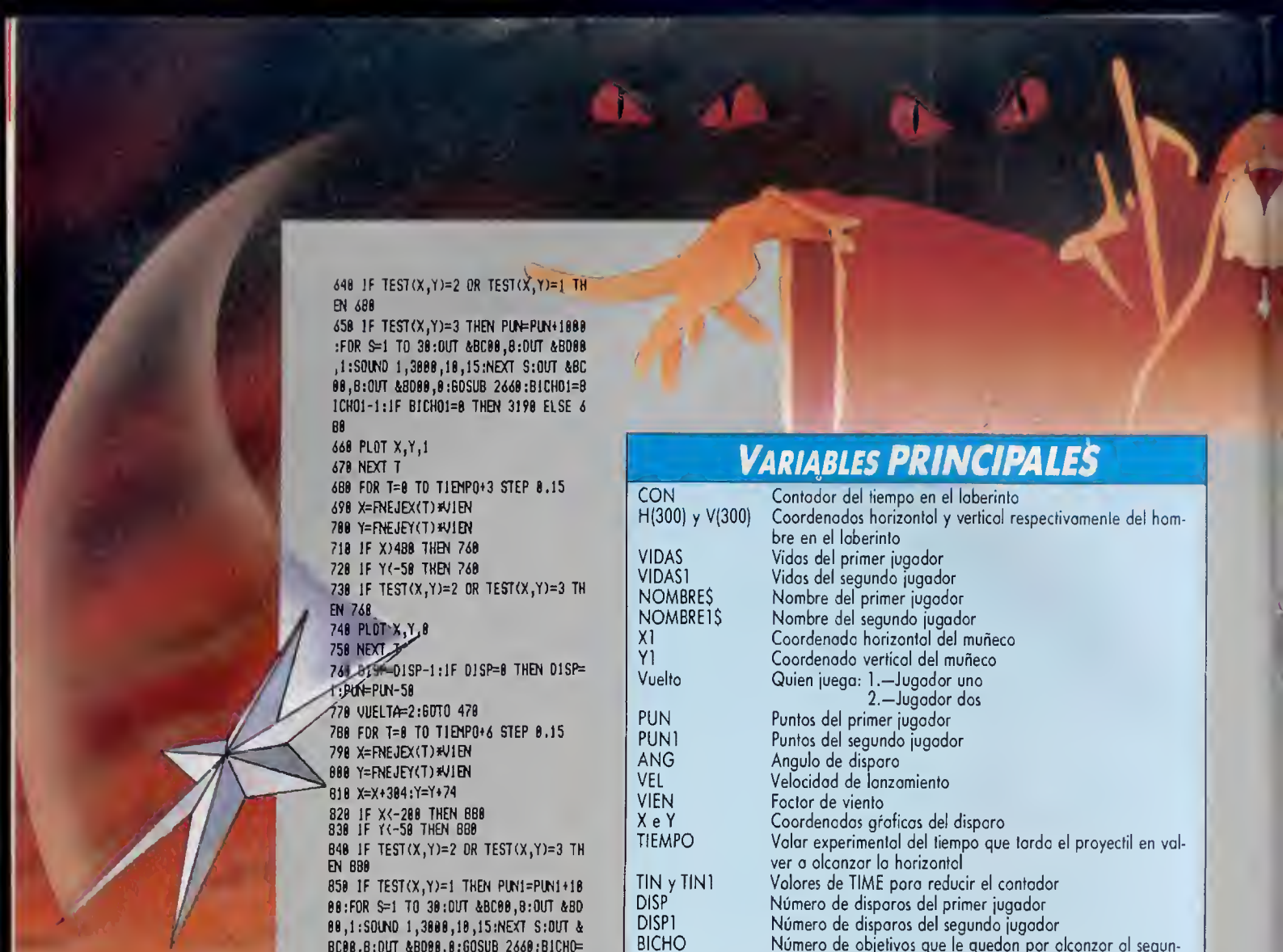
Castillo: aquí el turno de juego será alternativo. El propósito es intentar destruir los tres puntos vitales de tu adversario *(representados por tres cuadrados de color)*. Para poder disparar a tu contrario deberás dar al ordenador dos datos: el primero será el ángulo de disparo que deberá variar entre 0 y 90 y el segundo separado por una coma del primero, la velocidad que deberá variar entre 0 y 100. Para que el mismo ángulo y velocidad no le corresponda el mismo disparo el ordenador evalúa un factor de viento que desviará ligeramente el objetivo de nuestro disparo, haciendo casi imposible el que se produzcan dos disparos iguales.

En la parte inferior de la pantalla te aparecerá el nombre del jugador que en ese momento se halle en poder del turno de disparo y el número de disparos que le queda. Si ese número llega a 0 te dará siempre uno más, pero por contrapartida te quitará 50 puntos a tu marcador.

TABLA DE SUBRUTINAS

10-150	Pide nombres de las jugadores
160-250	Inicializa variables del laberinto
260-460	Dibuja castillo
470-980	Calcula coordenados del dispara de ambos jugadores
990-1080	Inicializo variables del laberinto
1090-1270	Dibuja el laberinto
1280-1370	Bucle de lectura del teclado
1380-1470	Detecta si ha habido chaque en el laberinto
1480-1520	Disminuye el contadar del tiempa
1530-1720	DATAS
1730-1810	Barra los pasas del muñeca y disminuye vidas
1820-1880	Came carazán
1890-2160	Sin vidas del jugadar 1
2170-2250	Fuera del laberinta
2260-2650	Instrucciones
2660-2740	Objetivo alcanzodo
2750-2990	Presentación
3000-3060	Código máquina
3070-3180	Pasa del laberinta al castillo
3190-3420	Fin de la partida
3430-3480	Sonido al salir fuera del laberinto

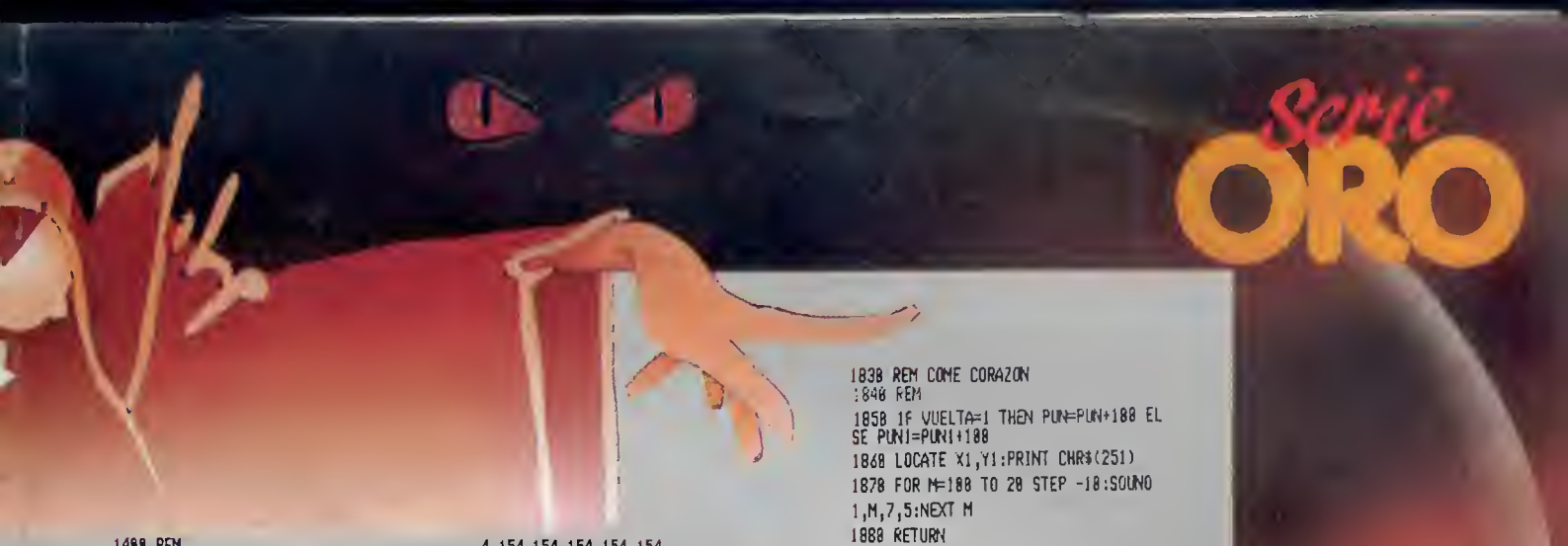
```
10 REM ********************************
***
20 REM *******  GRUPO DE ASALTO  *****
***
30 REM *******8y FERNANDO GARCIA *****
***
40 REM *******  and DANIEL CALVO *****
***
50 REM ********************************
***
60 MODE 1
70 GOSUB 3000:' C/M
80 GOTO 2750:' PRESENTACION
90 LOCATE 10,8:INPUT "JUGADOR 1..? "
,NOMBRE$
100 IF NOMBRE$="" THEN 90
110 IF LEN(NOMBRE$))8 THEN PRINT:PRI
NT TAB(10)"MAXIMO 8 LETRAS":GOTO 90
120 LOCATE 10,12:INPUT "JUGADOR 2..?
 ",NOMBRE1$
130 IF LEN(NOMBRE1$))8 THEN PRINT:PR
INT TAB(10)"MAXIMO 8 LETRAS":GOTO 12
0
140 NOMBRE$="SIR "+NOMBRE$:NOMBRE1$=
"SIR "+NOMBRE1$
150 GOTO 1020
160 REM
170 REM CASTILLO
180 REM
190 CALL &A000:INK 0,11:INK 1,24:INK
 2,12
200 PAPER 0
210 VUELTA=1:BICHO=3:BICHO1=3
220 PEN 2
230 CLS
240 DEF FNEJEX(T)=VEL*T*COS(ANG)
250 DEF FNEJEY(T)=VEL*T*SIN(ANG)-(4.
9*(T^2))
260 REM
270 REM CONSTRUIR DIBUJO
280 REM
290 ORIGIN 160,140
300 WINDOW #1,1,40,23,25
310 FOR X=1 TO 40:FOR Y=19 TO 21:LOC
ATE X,Y:PRINT CHR$(143);:NEXT Y:NEXT
 X
320 LOCATE 26,9:PRINT CHR$(222);CHR$
(223):LOCATE 25,10:PRINT CHR$(222);C
HR$(207);CHR$(207);CHR$(223)
330 LOCATE 35,9:PRINT CHR$(222);CHR$
(223):LOCATE 34,10:PRINT CHR$(222);C
HR$(207);CHR$(207);CHR$(223)
340 FOR S=25 TO 28:FOR D=11 TO 14:LO
CATE S,D:PRINT CHR$(250):NEXT D:NEXT
 S
350 FOR S=34 TO 37:FOR D=11 TO 14:LO
CATE S,D:PRINT CHR$(250):NEXT D:NEXT
 S
360 FOR S=25 TO 37:FOR D=15 TO 19:LO
CATE S,D:PRINT CHR$(250):NEXT D:NEXT
 S
370 LOCATE 4,16:PRINT STRING$(7,218)
:FOR X=4 TO 10:FOR A=17 TO 19:LOCATE
 X,A:PRINT CHR$(250):NEXT A:NEXT X
380 PLOT 0,0,2:DRAW 30,20
390 PLOT 368,60,2:DRAWR -30,30
400 PEN 1:LOCATE 1,10:PRINT CHR$(233
)
410 LOCATE 5,16:PRINT CHR$(233)
420 LOCATE 18,18:PRINT CHR$(233)
430 PEN 3:LOCATE 32,14:PRINT CHR$(23
3)
440 LOCATE 30,14:PRINT CHR$(233)
450 PLOT 270,114,2:DRAWR 10,0
460 LOCATE 28,9:PRINT CHR$(233)
470 REM
480 REM CALCULA VALORES
490 REM
500 ORIGIN 190,160
510 LOCATE #1,1,3:IF VUELTA=1 THEN P
RINT #1,NOMBRE$+" ";DISP ELSE PRINT
#1,NOMBRE1$+" ";DISP1
520 DEG:LOCATE #1,1,1:INPUT #1,"ANGU
LO (0-90),VELOCIDAD (0-100) ";ANG,VE
L
530 IF ANG)90 OR ANG<0 THEN 520
540 IF VEL<0 OR VEL)100 THEN 520
550 VIEN=0.7+(RND+1)/4
560 IF VUELTA=2 THEN ANG=180-ANG
570 TIEMPO=((VEL*SIN(ANG))/4.9)
580 IF VUELTA=1 THEN PLOT 0,0,0 ELSE
 PLOT 304,74,0:GOTO 780
590 FOR T=0 TO TIEMPO+3 STEP 0.15
600 X=FNEJEX(T)*VIEN
610 Y=FNEJEY(T)*VIEN
620 IF X)480 THEN 680
630 IF Y<-50 THEN 680
```



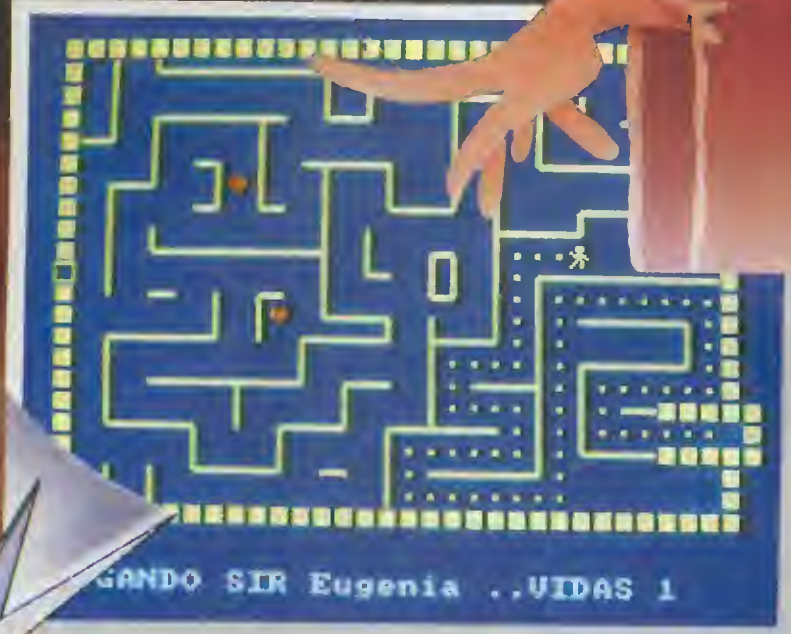
```
640 IF TEST(X,Y)=2 OR TEST(X,Y)=1 TH
EN 680
650 IF TEST(X,Y)=3 THEN PUN=PUN+1000
:FOR S=1 TO 30:OUT &BC00,8:OUT &BD00
,1:SOUND 1,3000,10,15:NEXT S:OUT &BC
00,8:OUT &BD00,0:GOSUB 2660:BICHO1=B
ICHO1-1:IF BICHO1=0 THEN 3190 ELSE 6
80
660 PLOT X,Y,1
670 NEXT T
680 FOR T=0 TO TIEMPO+3 STEP 0.15
690 X=FNEJEX(T)*VIEN
700 Y=FNEJEY(T)*VIEN
710 IF X)480 THEN 760
720 IF Y(-50 THEN 760
730 IF TEST(X,Y)=2 OR TEST(X,Y)=3 TH
EN 760
740 PLOT X,Y,0
750 NEXT T
760 DISP=DISP-1:IF DISP=0 THEN DISP=
1:PUN=PUN-50
770 VUELTA=2:GOTO 470
780 FOR T=0 TO TIEMPO+6 STEP 0.15
790 X=FNEJEX(T)*VIEN
800 Y=FNEJEY(T)*VIEN
810 X=X+304:Y=Y+74
820 IF X(-200 THEN 880
830 IF Y(-50 THEN 880
840 IF TEST(X,Y)=2 OR TEST(X,Y)=3 TH
EN 880
850 IF TEST(X,Y)=1 THEN PUN1=PUN1+10
00:FOR S=1 TO 30:OUT &BC00,8:OUT &BD
00,1:SOUND 1,3000,10,15:NEXT S:OUT &
BC00,8:OUT &BD00,0:GOSUB 2660:BICHO=
BICHO-1:IF BICHO=0 THEN 3190 ELSE 88
0
860 PLOT X,Y,3
870 NEXT T
880 FOR T=0 TO TIEMPO+6 STEP 0.15
890 X=FNEJEX(T)*VIEN
900 Y=FNEJEY(T)*VIEN
910 X=X+304:Y=Y+74
920 IF X(-200 THEN 970
930 IF Y(-50 THEN 970
940 IF TEST(X,Y)=1 OR TEST(X,Y)=2 TH
EN 970
950 PLOT X,Y,0
960 NEXT T
970 DISP1=DISP1-1:IF DISP1=0 THEN DI
SP1=1:PUN1=PUN1-50
980 VUELTA=1:GOTO 470
990 REM
1000 REM PRIMERA FASE .....
1010 REM
1020 REM
1030 REM VARIABLES LABERINTO
1040 REM
1050 VUELTA=1:VIDAS=5:VIDAS1=5
1060 CON=500:DIM H(300),V(300):Z=2:X
1=32:Y1=18:H(1)=X1:V(1)=Y1:N=1
1070 ORIGIN 0,0:INK 0,1:INK 1,1:INK
2,1:INK 3,1:PEN 1:BORDER 1
1080 CLS
1090 REM
1100 REM DIBUJA LABERINTO
1110 REM
1120 LOCATE 1,1:PRINT STRING$(32,233
):LOCATE 1,22:PRINT STRING$(32,233)
1130 FOR D=1 TO 22:LOCATE 1,D:PRINT
CHR$(233):LOCATE 32,D:PRINT CHR$(233
):NEXT D
1140 LOCATE 32,5:PRINT " ":LOCATE 32
,18:PRINT CHR$(251)
1150 LOCATE 33,17:PRINT CHR$(233);CH
R$(10);CHR$(8);CHR$(233);CHR$(10);CH
R$(8);CHR$(233)
1160 RESTORE 1530
1170 FOR C=2 TO 31
1180 READ A
1190 IF A=220 THEN PEN 3:LOCATE C,2:
PRINT CHR$(A):PEN 1:GOTO 1210
1200 LOCATE C,2:PRINT CHR$(A)
1210 NEXT C
1220 Z=Z+1:IF Z)21 THEN 1230 ELSE 11
70
1230 PLOT 2,35,1:DRAW CON,35:PLOT 2,
36:DRAW CON,36
1240 PEN 2:LOCATE 1,25:PRINT "JUGANO
O ";:IF VUELTA=1 THEN PRINT NOMBRE$;
" ..VIDAS";VIDAS;"    " ELSE PRINT N
OMBRE1$;" ..VIDAS";VIDAS1;"    "
1250 INK 1,24:INK 2,28:INK 3,6
1260 PEN 1
1270 TIN=INT(TIME/300)
1280 REM
1290 REM MOVIMIENTO DEL HOMBRE
1300 REM
1310 IF INKEY(71)=0 THEN LOCATE X1,Y
1:PRINT CHR$(144):Y1=Y1+1:GOSUB 1380
1320 IF INKEY(39)=0 THEN LOCATE X1,Y
1:PRINT CHR$(144):X1=X1-1:GOSUB 1380
1330 IF INKEY(31)=0 THEN LOCATE X1,Y
1:PRINT CHR$(144):X1=X1+1:GOSUB 1380
1340 IF INKEY(69)=0 THEN LOCATE X1,Y
1:PRINT CHR$(144):Y1=Y1-1:GOSUB 1380
1350 TIN1=INT(TIME/300)
1360 IF TIN1=TIN+1 THEN GOTO 1400
1370 GOTO 1280
1380 REM
1390 REM HAY CHOQUE ??
```

VARIABLES PRINCIPALES

CON	Contador del tiempo en el laberinto
H(300) y V(300)	Coordenadas horizontal y vertical respectivamente del hombre en el laberinto
VIDAS	Vidas del primer jugador
VIDAS1	Vidas del segundo jugador
NOMBRE$	Nombre del primer jugador
NOMBRE1$	Nombre del segundo jugador
X1	Coordenada horizontal del muñeco
Y1	Coordenada vertical del muñeco
Vuelto	Quien juega: 1.—Jugador uno 2.—Jugador dos
PUN	Puntos del primer jugador
PUN1	Puntos del segundo jugador
ANG	Angulo de disparo
VEL	Velocidad de lanzamiento
VIEN	Factor de viento
X e Y	Coordenadas gráficas del disparo
TIEMPO	Valor experimental del tiempo que tarda el proyectil en volver a alcanzar la horizontal
TIN y TIN1	Valores de TIME para reducir el contador
DISP	Número de disparos del primer jugador
DISP1	Número de disparos del segundo jugador
BICHO	Número de objetivos que le quedan por alcanzar al segundo jugador
BICHO1	Número de objetivos que le quedan por alcanzar al primer jugador
FINEJEX(t)	Fórmula física para el cálculo de lanzamiento de proyectiles en el eje x
FNEJEY(t)	Fórmula física para el cálculo de lanzamiento de proyectiles en el eje y



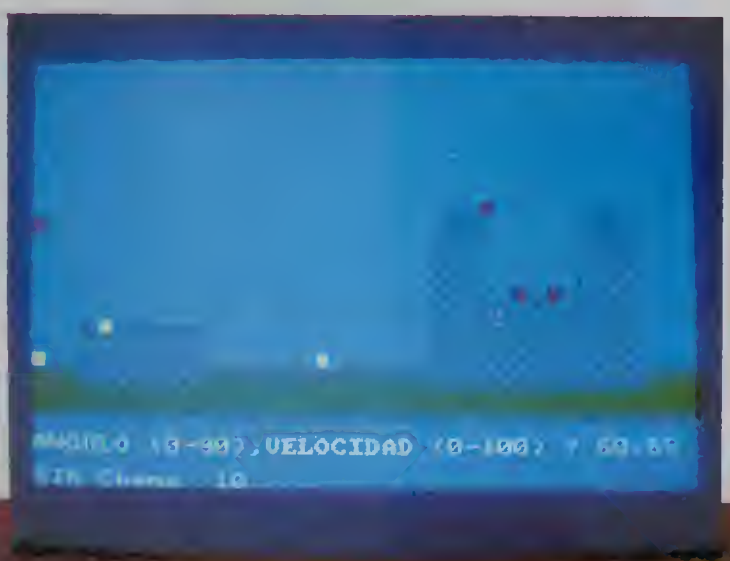

```
1400 REM
1410 IF TEST(X1*16-8,486-(Y1*16))=1
OR TEST(X1*16-18,486-(Y1*16))=1 THEN
 1730
1420 IF TEST(X1*16-18,486-(Y1*16))=3
 THEN GOSUB 1820
1430 LOCATE X1,Y1:PRINT CHR$(251)
1440 IF X1)32 THEN 2170
1450 FOR f=1 TO 30:NEXT f
1460 N=N+1:H(N)=X1:V(N)=Y1
1470 RETURN
1480 REM
1490 REM DISMINUYE TIEMPO
1500 REM
1510 PLOT CON,35,0:CON=CON-15:DRAW C
ON,35:PLOT CON+15,36:DRAW CON,36:IF
CON<=2 THEN 1740
1520 GOTO 1270
1530 DATA 128,149,128,147,154,154,15
4,154,154,154,154,156,128,128,128,12
8,128,128,128,128,128,128,128,128,12
8,128,128,128,128,228
1540 DATA 128,149,128,128,128,128,12
8,128,128,128,128,149,128,148,128,14
6,154,158,154,154,154,156,128,148,12
8,128,128,128,128,128
1550 DATA 128,149,128,158,154,154,15
4,154,158,152,128,147,154,153,128,12
8,128,149,128,128,128,149,128,145,12
8,146,154,233,233,233
1560 DATA 154,153,128,149,128,128,12
8,128,149,128,128,128,128,128,128,12
8,128,149,128,148,128,149,128,128,12
8,128,128,128,128,128
1570 DATA 128,128,128,149,128,146,15
6,128,149,128,146,158,154,154,156,12
8,128,149,128,149,128,147,154,154,15
4,154,154,233,233,233
1580 DATA 128,158,154,153,128,128,14
9,228,149,128,128,149,128,128,149,12
8,128,149,128,149,128,128,128,128,12
8,128,128,128,128,128
1590 DATA 128,149,128,128,128,146,15
3,128,147,152,128,149,128,128,147,15
4,154,153,128,149,128,128,128,158,15
4,154,154,154,128,128
1600 DATA 128,149,128,148,128,128,12
8,128,128,128,128,149,128,148,128,12
8,128,128,128,151,154,154,154,153,12
8,128,128,128,128,128
1610 DATA 128,149,128,147,154,154,15
4,154,154,154,154,157,128,149,128,12
8,158,156,128,149,128,128,128,128,12
8,128,128,128,128,128
1620 DATA 128,149,128,128,128,128,12
8,128,128,128,128,149,128,147,152,12
8,149,149,128,149,128,158,154,154,15
4,154,154,154,154,154
1630 DATA 128,145,128,146,152,128,14
8,128,158,152,128,149,128,128,128,12
8,147,153,128,149,128,149,128,128,12
8,128,128,128,128,128
1640 DATA 128,128,128,128,128,128,14
9,128,149,228,128,147,154,154,156,12
8,148,128,128,149,128,149,128,146,15
4,154,154,154,156,128
1650 DATA 128,158,154,154,154,154,15
3,128,145,128,158,154,154,154,153,12
8,151,154,154,153,128,149,128,128,12
8,128,128,128,149,128
1660 DATA 128,149,128,128,128,128,12
8,128,128,128,149,128,128,128,128,12
8,149,128,128,128,128,149,128,158,15
4,154,154,154,153,128
1670 DATA 128,149,128,146,154,154,15
4,158,154,154,153,128,158,154,152,12
8,149,128,146,154,154,157,128,149,12
8,128,128,128,128,128
1680 DATA 128,149,128,128,128,128,12
8,149,128,128,128,128,149,128,128,12
8,149,128,128,128,128,149,128,149,12
8,146,154,233,233,233
1690 DATA 128,147,154,154,154,156,12
8,145,128,158,154,154,153,128,158,15
4,155,154,154,152,128,149,128,149,12
8,128,128,128,128,128
1700 DATA 128,128,128,128,128,149,12
8,128,128,149,128,128,128,128,149,12
8,128,128,128,128,128,149,128,147,15
4,154,154,233,233,233
1710 DATA 128,148,128,148,128,147,15
4,154,154,153,128,146,152,128,149,12
8,146,154,154,154,154,153,128,128,12
8,128,128,128,128,128
1720 DATA 128,149,128,149,128,128,12
8,128,128,128,128,128,128,128,149,12
8,128,128,128,128,128,128,128,128,12
8,128,128,128,128,128
1730 REM
1740 REM BORRADO Y DISMINUYE VIDAS
1750 REM
1760 IF VUELTA=2 THEN VIDAS1=VIDAS1-
1:IF VIDAS1=0 THEN GOTO 2170
1770 IF VUELTA=1 THEN VIDAS=VIDAS-1:
IF VIDAS=0 THEN VUELTA=VUELTA+1:ERAS
E H,V:GOTO 1890
1780 FOR S=1 TO N:LOCATE H(S),V(S):P
RINT " ":PRINT CHR$(7):NEXT S
1790 X1=31:Y1=18:LOCATE X1,Y1:PRINT
CHR$(251):H(1)=X1:V(1)=Y1:N=1:CON=58
0
1800 GOTO 1230
1810 RETURN
1820 REM
1830 REM COME CORAZON
:1840 REM
1850 IF VUELTA=1 THEN PUN=PUN+100 EL
SE PUN1=PUN1+100
1860 LOCATE X1,Y1:PRINT CHR$(251)
1870 FOR M=100 TO 20 STEP -10:SOUND
1,M,7,5:NEXT M
1880 RETURN
1890 REM
1900 REM SIN VIDAS
1910 REM
1920 CALL &A800:CLS:LOCATE 5,3:PRINT
 CHR$(150)+STRING$(30,154)+CHR$(156)
1930 LOCATE 5,4:PRINT CHR$(149)+STRI
NG$(30,32)+CHR$(149)
1940 LOCATE 5,5:PRINT CHR$(149)+STRI
NG$(30,32)+CHR$(149)
1950 LOCATE 5,6:PRINT CHR$(149)TAB(1
5)+"LASTIMA!!!"+STRING$(11,32)+CHR$(
149)
1960 LOCATE 5,7:PRINT CHR$(149)TAB(3
6)CHR$(149)
1970 LOCATE 5,8:PRINT CHR$(149)+"  N
O CONSEGUISTES PASAR EL    "+CHR$(14
9)
1980 LOCATE 5,9:PRINT CHR$(149)+STRI
NG$(30,32)+CHR$(149)
1990 LOCATE 5,10:PRINT CHR$(149)+STR
ING$(9,32)+"LABERINTO."+STRING$(11,3
2)+CHR$(149)
2000 LOCATE 5,11:PRINT CHR$(149)TAB(
36)CHR$(149)
2010 LOCATE 5,12:PRINT CHR$(149)TAB(
36)CHR$(149)
2020 LOCATE 5,13:PRINT CHR$(149)+"
TUS PUNTOS HAN SIDO ..";:PEN 2:PRINT
 USING "####";pun;:PEN 1:PRINT "  "+
CHR$(149)
2030 LOCATE 5,14:PRINT CHR$(149)TAB(
36)CHR$(149)
2040 LOCATE 5,15:PRINT CHR$(149)+"
```




```
ES EL TURNO DE ";:PEN 2:PRINT NOMBRE
1$;:PEN 1:PRINT TA8(36)CHR$(149)
2050 LOCATE 5,16:PRINT CHR$(149)TA8(
36)CHR$(149)
2060 LOCATE 5,17:PRINT CHR$(149)TA8(
36)CHR$(149)
2070 LOCATE 1,18:PRINT CHR$(150)+CHR
$(154)+CHR$(154)+CHR$(154)+CHR$(153)
+STRING$(38,32)+CHR$(147)+CHR$(154)+
CHR$(154)+CHR$(154)+CHR$(156)
2080 LOCATE 1,19:PRINT CHR$(149)TA8(
40)CHR$(149)
2090 LOCATE 1,20:PRINT CHR$(151)+STR
ING$(38,154)+CHR$(157)
2100 LOCATE 1,21:PRINT CHR$(149)TA8(
40)CHR$(149)
2110 LOCATE 1,22:PRINT CHR$(149)+STR
ING$(9,32);:PEN 2:PRINT"(PULSA UNA T
ECLA)"+STRING$(12,32);:PEN 1:PRINT C
HR$(149)
2120 LOCATE 1,23:PRINT CHR$(149)TA8(
40)CHR$(149)
2130 LOCATE 1,24:PRINT CHR$(147)+STR
ING$(38,154)+CHR$(153)
2140 CALL &A000
2150 K$="":WHILE K$="":K$=INKEY$:WEN
D
2160 VUELTA=2:GOTO 1060
2170 REM
2180 REM FUERA DEL LABERINTO
2190 REM .
2200 IF VUELTA=2 AND VIDAS1)0 THEN P
UN1=PUN1+VIDAS1*100+CON*3+N*10:GOSUB
 3430
2210 IF VUELTA=1 THEN PUN=PUN+VIDAS*
100+CON*3+N*10:VUELTA=VUELTA+1:ERASE
H,V:GOSUB 3430:GOTO 1060
2220 IF VIDAS)0 THEN DISP=INT(PUN/80
) ELSE DISP=10
2230 IF VIDAS1)0 THEN DISP1=INT(PUN1
/80) ELSE DISP1=10
2250 GOTO 3070
2260 REM
2270 REM INSTRUCCIONES
2280 REM
2290 CLS
2300 LOCATE 14,2:PRINT "AMS-CASTILLO
"
2310 LOCATE 13,3:PRINT STRING$(14,20
0)
2320 LOCATE 6,4:PRINT "Estais en ple
no siglo XIV. Las"
2330 LOCATE 3,6:PRINT "cosas no os v
an demasiado bien. El"
2340 LOCATE 3,8:PRINT "malvado rey h
a ordenado que sus dos"
2350 LOCATE 3,10:PRINT "mejores caba
lleros se enfrenten en"
2360 LOCATE 3,12:PRINT "un combate a
 muerte tras sobrevivir"
2370 LOCATE 3,14:PRINT "a la dura pr
ueba del laberinto."
2380 LOCATE 6,16:PRINT "Pocos han si
do los valientes que"
2390 LOCATE 3,18:PRINT "han logrado
pasarla, en una lucha"
2400 LOCATE 3,20:PRINT "contra el ti
empo y la codicia."
2410 LOCATE 22,22:PRINT CHR$(150)+ST
RING$(15,154)+CHR$(156)
2420 LOCATE 22,23:PRINT CHR$(149)+"P
ULSA UNA TECLA"+CHR$(149)
2430 LOCATE 22,24:PRINT CHR$(147)+ST
RING$(15,154)+CHR$(153)
2440 CALL &A000
2450 K$="":WHILE K$="":K$=INKEY$:WEN
D
2460 CLS
2470 LOCATE  6, 3:PRINT "Para poder
ganar a tu adversario"
2480 LOCATE  4, 5:PRINT "deberas des
truir todos sus puntos"
2490 LOCATE  4, 7:PRINT "vitales, ya
 sea en el castillo o en"
2500 LOCATE  4, 9:PRINT "el bunker.
Tened cuidado, vuestros"
2510 LOCATE  4, 11:PRINT "puntos dep
enderan de las acciones"
2520 LOCATE  4, 13:PRINT "que empren
dais a lo largo del juego."
2530 LOCATE  6, 15:PRINT "El viento
y vuestra punteria deci-"
2540 LOCATE  4, 17:PRINT "diran la p
artida."
2550 LOCATE  20, 18:PRINT CHR$(150)
2560 LOCATE  20, 19:PRINT CHR$(149)+
" "+CHR$(240)+"[A] "+CHR$(241)+"[Z]
"+CHR$(242)+"[<] "+CHR$(243)+"[>]"
2570 LOCATE  5, 20:PRINT "Movimiento
s:   "+CHR$(157)
2580 LOCATE  20, 21:PRINT CHR$(149)+
" -INTRODUCIR DATOS"
2590 LOCATE  3,22:PRINT CHR$(150)+ST
RING$(15,154)+CHR$(156):LOCATE 20, 2
2:PRINT CHR$(147)
2600 LOCATE  3,23:PRINT CHR$(149)+"P
ULSA UNA TECLA"+CHR$(149)
2610 LOCATE  3,24:PRINT CHR$(147)+ST
RING$(15,154)+CHR$(153)
2620 CALL &A000
2630 K$="":WHILE K$="":K$=INKEY$:WEN
D
2640 CLS
2650 GOTO 90
2660 REM
2670 REM OBJETIVO ALCANZADO
2680 REM
2690 IF X(-170 THEN LOCATE 1,18:PRIN
T " ":RETURN
2700 IF X(-100 THEN LOCATE 5,16:PEN
2:PRINT CHR$(210):RETURN
2710 IF X(100 THEN LOCATE 18,18:PRIN
T " ":RETURN
2720 IF X)300 THEN LOCATE 32,14:PRIN
T " ":RETURN
2730 IF X)270 THEN LOCATE 30,14:PRIN
T " ":RETURN
2740 IF X)240 THEN LOCATE 28,9:PRINT
 " ":RETURN
2750 REM
2760 REM PRESENTACION
2770 REM
2780 CLS
2790 INK 0,0:INK 1,0:INK 2,23:BORDER
 0:PEN 1:FOR T=1 TO 200:NEXT T
2800 LOCATE 1,25:PRINT "AMS.CASTILLO
"
2810 FOR X=1 TO 90 STEP 2
2820 FOR Y=16 TO 1 STEP -2
2830 IF TEST(X,Y)()0 THEN PLOT -3+X*
3.3,340+(Y*4)-X*2,2:DRAWR 6,6:DRAWR
-6,-6:DRAWR 12,3,3
```

```
2040 NEXT Y
2050 NEXT X
2060 FOR X=98 TO 192 STEP 2
2070 FOR Y=16 TO 1 STEP -2
2080 IF TEST(X,Y)<>0 THEN PLOT 16+X*
3.3,-46+(Y*4)+X*2,2:DRAWR 6,6:DRAWR
-12,3,3
2090 NEXT Y
2900 NEXT X
2910 LOCATE 1,25:PEN 0:PRINT "

2920 INK 1,26:PEN 1
2930 CALL &A000
2940 LOCATE 5,23:INPUT "QUIERES INST
RUCCIONES ";RES$
2950 IF RES$="" THEN 2940
2960 IF LEFT$(UPPER$(RES$),1)="S" TH
EN GOTO 2260
2970 IF LEFT$(UPPER$(RES$),1)="N" TH
EN CLS:GOTO 90
2990 GOTO 2940
3000 REM
3010 REM CODIGO MAQUINA
3020 REM
3030 RESTORE 3040:FOR X=1 TO 6:READ
A$:POKE &9FFF+X,VAL("&"+A$):NEXT X
3040 DATA CD,09,BB,38,F8,C9
3050 SYMBOL 250,&FF,&8,&8,&8,&FF,&00
,&00,&00
3060 RETURN
3070 REM
3080 REM PASO DEL ECUADOR
3090 REM
3100 CLS
3110 LOCATE 2,8:PRINT NOMBRE$:LOCATE
 2,16:PRINT NOMBRE1$
3120 LOCATE 16,2:PRINT "PUNTOS":LOCA
TE 25,2:PRINT "DISPARDS"
3130 PEN 2
3140 LOCATE 18,8:PRINT USING "####";
PUN:LOCATE 31,8: PRINT USING "##";DI
SP
3150 LOCATE 18,16:PRINT USING "####"
;PUN1:LOCATE 31,16: PRINT USING "##"
;DISP1
3160 CALL &A000
3170 WHILE INKEY$="":WEND
3180 GOTO 160
3190 REM
3200 REM FIN!!! POR FIN!!!
3210 REM
3220 CLS
3230 PEN 1
3240 INK 0,1
3250 LOCATE 8,4:PRINT "HAS LOGRADO S
OBREVIVIR"
3260 IF BICHO1=0 THEN PRINT:PRINT TA
B(8)NOMBRE$:PUN=PUN+DISP*50 ELSE PRI
NT:PRINT TAB(8)NOMBRE1$:PUN1=PUN1+DI
SP1*50
3270 LOCATE 8,8:PRINT "Y HAS OBTENID
O UN TOTAL"
3280 LOCATE 8,10:PRINT "DE";
3290 IF BICHO1=0 THEN LOCATE 18,10:P
RINT PUN; ELSE LOCATE 18,10:PRINT PU
N1;
3300 PRINT " PUNTOS."
3310 LOCATE 8,12:PRINT "TU ENEMIGO "
;
3320 IF BICHO1=0 THEN PRINT NOMBRE1$
 ELSE PRINT NOMBRE$
3330 LOCATE 8,14:PRINT "HA CONSEGUID
O";
3340 IF BICHO1=0 THEN PRINT PUN1; EL
SE PRINT PUN;
3350 PRINT " PUNTOS"
3360 LOCATE 13,16:PRINT "!!ENHORABUE
NA!!!
3370 PRINT
3380 IF BICHO1=0 THEN PRINT TAB(10)"
LORD ";NOMBRE$ ELSE PRINT TAB(10)"LO
RD ";NOMBRE1$
3390 LOCATE 10,25:PRINT "PULSA UNA T
ECLA"
3400 CALL &A000
3410 K$="":WHILE K$="":K$=INKEY$:WEN
D
3420 RUN
3430 REM
3440 REM SONIDO
3450 REM
3460 SOUND 1,478,30,15:SOUND 1,358,9
0,15:SOUND 1,319,30,15::SOUND 1,301,
30,15:SOUND 1,319,30,15::SOUND 1,301
,30,15:SOUND 1,358,30,15:SOUND 1,402
,90,15:SOUND 1,478,30,15:SOUND 1,402
,90,15
3470 SOUND 1,478,30,15:SOUND 1,358,9
0,15:SOUND 1,319,30,15::SOUND 1,301,
30,15:SOUND 1,319,30,15::SOUND 1,301
,30,15:SOUND 1,268,30,15:SOUND 1,239
,90,15:SOUND 1,301,30,15:SOUND 1,239
,90,15
3480 RETURN
```



AMSTRADIDEAS

EL 464 ES UN 664 DISFRAZADO

Por Andrés del Pozo Prieto

Los afortunados poseedores de un CPC 464 probablemente os habréis encontrado en multitud de ocasiones con comandos tales como FRAME, CURSOR, GRAPHICS PAPER, etc. procedentes del BASIC 1.1 que utilizan el 664 y 6128. Normalmente cuando queréis copiar un listado con estos comandos los tenéis que anular *(con la consiguiente pérdida de efectos gráficos)* o cargar antes un programa que los sustituya por comandos RSX *(como el del Microhobby* **AMSTRAD** *n.º 42)* con el engorro adicional de tener que teclearlo y después cargarlo cada vez que se quiera correr el programa. Esto para los que poseen una unidad de disco no supone ningún problema pero con el cassette con que viene equipado el 464 es un verdadero laberinto.

LA OTRA OPCION

Existe, sin embargo, otros métodos para emular algunos de los comandos del BASIC 1.1 basados en llamadas directas a las rutinas del Firmware o mediante cortas líneas Basic:

— FRAME: Este comando es sencillísimo de sustituir tecleando: CALL & BB19

— AEAR INPUNT: Para imitarlo se debe teclear la línea BASIC: WHILE INKEY$ < + '''': WEND.

— GRAPHICS PAPER: Este comando requiere más pericia para descubrir cómo imitarlo sin tener que recurrir al código máquina. Para ello hay que hacer una llamada al Firmware en la dirección & BBE4 y detrás de esta llamada hay que poner tantos parámetros como número de paper se quiera.

Por ejemplo para sustituir GRAPHICS PAPER 3 tenemos que escribir en nuestro 464 CALL & BBE4,1,1,1. Si queremos utilizar el papel 4 escribiremos CALL & BBE4,1,1,1,1.

Esto por supuesto en el caso del papel 15 sería **«algo»** incómodo.

— GRAPHICS PEN: El sistema utilizado que para GRAPHICS PAPER pero llamando a la dirección & BBDE.

— CURSOR: Cuando se quiere activar el cursor *(comando CURSOR 1,1)* hay que teclear: CALL & BB7B: CALL & BB81.

Para desactivarlo se escribe: CALL & BB7E *(CURSOR 1,0)* o CALL & BB84 *(CURSOR 0,1)*.

— Para seleccionar el modo lógico en que se van a imprimir los gráficos, en el BASIC 1.1 se utiliza un cuarto parámetro añadido a los comandos PLOT, DRAW; MOVE etc.

En el BASIC 1.0 (el del 464) podemos utilizar los caracteres de control (ver el manual), en concreto el n.º 23. Para seleccionar el modo gráfico se escribe: ? CHR$ (23); CHR$ (modo gráfico).

Los modos gráficos disponibles son los mismos que los del BASIC 1.1 y son los siguientes:

0-Modo normal

1-Modo XOR. Ejecuta un XOR entre la tinta ya impresa en la pantalla y la tinta que queremos poner.

2-Modo AND. Igual que el modo 1 pero efectuando un AND.

3-Modo OR. Igual que el modo 1 pero efectuando un OR.

Por desgracia los otros tres comandos no expuestos del BASIC 1.1 *(FILL, COPYCHR$, MASK)* son imposibles de imitar sin utilizar rutinas en código máquina como la del M.H.A. n.º 42.

Espero que estos pequeños truquitos os sean de gran utilidad al crear maravillosos gráficos con vuestro 464 y su magnífico Firmware.

AMSTRAD ESPECIAL N.º 3
DESDE HOY EN TU KIOSCO

EL FUTURO EMPEZO AYER

Descubre el fascinante mundo de los robots: Su utilidad, los últimos avances y cómo afectarán a tu vida. ● El juego de la vida: nacimiento, evolución y muerte en un mundo de fantasía. ● Simuladores de vuelo: Un trepidante vuelo sobre todos los simuladores de combate para AMSTRAD. ● HISOFT-C: El lenguaje de alto nivel más rápido del mundo para tu CPC. ● El mundo de bloques: Compresión del lenguaje natural desde Basic. ● Forth: Todo un lenguaje especialmente diseñado para ti. ● Todo esto y mucho más en AMSTRAD ESPECIAL NUMERO TRES.

Si no lo encontraras en tu kiosco, puedes solicitarlo directamente a nuestra editorial, llamando por teléfono (91) 734 65 00, o bien, rellenando el cupón.

Recorta o copia este cupón y envíalo a Hobby Press, Apartado de Correos 232. Alcobendas (Madrid)

SI deseo recibir en mi domicilio el especial de Amstrad n.º 3 al precio de 350 ptas.

APELLIDOS
NOMBRE
DOMICILIO
LOCALIDAD
TELEFONO
FECHA DE NACIMIENTO
PROVINCIA
CODIGO POSTAL
¿ERES SUSCRIPTOR DE MICROHOBBY AMSTRAD?

Forma de pago:
☐ Mediante talón bancario adjunto a nombre de Hobby Press, S. A.
☐ Mediante giro postal n.º .....
☐ Mediante tarjeta de crédito n.º □□□□ □□□□ □□□□ □□□□
☐ Visa ☐ Master Charge ☐ American Express
Fecha de caducidad de la tarjeta
☐ Contra reembolso del envío (suponen 100 ptas. más de gastos de envío)

Fecha y firma

DIFERENTES TAMAÑOS DE LETRAS EN UN MISMO MODO

Hace unos días vimos en estas mismas páginas, la forma de obtener diferentes tipos de caracteres a través de diferentes rutinas; hoy para exprimir aún más, si cabe, la capacidad de dichas rutinas, hemos preparado un programa que nos permitirá imprimir dichos caracteres a tres tamaños diferentes.

ntes que nada, debemos decir que el programa que os presentamos hoy funciona correctamente tanto si deseamos utilizarlo solo o acompañado del programa generador de caracteres.

Veamos ahora cuáles son las posibilidades que nos ofrece la rutina magnificadora de caracteres, y al finalizar el artículo explicaremos los pasos que deberemos seguir para utilizarlo solo o bien acompañado de nuestro generador de caracteres.

El programa nos ofrece tres nuevos comandos RSX que podremos utilizar desde nuestros propios programas Basic, éstos son los siguientes:

```
|T1,X,Y,"..."
|T2,X,Y,"..."
|T3,X,Y,"..."
```

donde X e Y deberán ser las coordenadas de impresión en pantalla, del mismo modo que se utilizan con la función 'LOCATE' del Basic, es decir, 'X' contiene la coordenada horizontal e 'Y' la coordenada vertical.

El último parámetro deberá contener la cadena de caracteres que se desea imprimir en la anterior posición de pantalla, y que no deberá superar los 80 caracteres de longitud.

Los tres nuevos comandos funcionan de una forma similar en cuanto a la impresión en pantalla, aunque poseen rutinas diferentes para manipular los caracteres que se deseen imprimir en pantalla.

Cuando se entra en la rutina, lo primero que se hace es comprobar que los parámetros introducidos con el comando RSX son correctos, de lo contrario se nos mostrará un mensaje de error y retornará al Basic.

Si los parámetros no son erróneos, se tomará la cadena de caracteres y se traspasará a un buffer propio para poderlos leer más tarde.

Una vez hecho esto, se tomará cada uno de los caracteres y se seguirán los siguientes pasos:

1.—Se toma la matriz donde se encuentran los datos que definen el carácter.

2.—Se modifican según el tipo de letra que se desee obtener.

3.—Se imprimen en pantalla.

Los datos modificados del carácter, se introducen en la matriz de los caracteres gráficos 252-255, para más tarde imprimirlos en pantalla.

Veamos cómo funcionan las diferentes rutinas encargadas de la transformación de los caracteres gráficos.

El primero de los comandos se encarga de imprimir en pantalla un tipo de letra dos veces más largo que el presentado normalmente en el modo de pantalla en que se esté trabajando en ese momento.

Para ello, se tomará la matriz del carácter a imprimir, y se doblarán cada uno de los bytes que lo definen de la forma siguiente:

```
00011000        00011000
00011000        00011000
00100100        00011000
01000010        00100100
01000010        00100100
01111110        01000010
01000010        01000010
01000010        01000010
00000000        01000010
                01111110
                01111110
                01000010
                01000010
```

```
01000010
01000010
00000000
00000000
```

Con lo cual se habrá conseguido un carácter dos veces más alto que el original.

El segundo comando 'T2', nos proporciona un carácter con un ancho doble que el original, ello se consigue doblando cada uno de los bits que componen los ocho bytes que defienen el carácter original.

Para ello se llama a una rutina que debe producir el efecto que mostramos a continuación, tomando como carácter original el mismo carácter tomado anteriormente «A»:

```
0000001111000000
0000110000110000
0011000000001100
0011000000001100
0011111111111100
0011000000001100
0011000000001100
0000000000000000
```

Es decir, se deben doblar cada uno de los bits que componen los ocho bytes que definen el carácter original. Así pues al finalizar la operación habremos obtenido 16 bytes que son los que definirán el carácter ampliado.

El tercer tipo de carácter, se consigue aplicando los dos efectos anteriores, con lo que se conseguirán 32 bytes que serán los que nos proporcionarán la definición del carácter de doble ampliación.

El efecto producido sería el que se muestra gráficamente a continuación:

```
0000001111000000
0000001111000000
0000110000110000
0000110000110000
0011000000001100
0011000000001100
0011000000001100
0011000000001100
0011111111111100
0011111111111100
0011000000001100
0011000000001100
0011000000001100
0011000000001100
0000000000000000
0000000000000000
```

Veamos a continuación en qué ocasiones nos puede aparecer el siguiente mensaje de error:

ERROR EN PARAMETROS

—Cuando se omita alguno de los tres parámetros necesarios para el buen funcionamiento del comando.

—Cuando la longitud de la cadena introducida sea superior a 80 caracteres.

Veamos cuáles son los pasos a seguir para almacenar la presente rutina en cinta o disco.

Para aquellos que posean ensamblador, únicamente tienen que copiar el listado desensamblado que aparece a continuación, y salvarlo como código objeto a partir de la dirección hexadecimal #9000 con una longitud de 501 bytes.

Si se prefiere se puede utilizar el cargador Basic, para ello, una vez copiado se deberá ejecutar, y en el caso de que aparezca algún mensaje de error se deberán revisar las líneas 'DATA'.

Si se ha ejecutado correctamente, procederemos a salvar el programa de la siguiente forma:

```
SAVE "CARAC2",B,&9000,501
```

Cuando se desee ejecutar, deberemos cargar la rutina almacenada anteriormente, e inicializar los comando RSX. Esto se puede realizar de la siguiente forma:

```
10 MEMORY &8FFF
20 LOAD "CARAC2", &9000
30 CALL &9000
```

De esta forma ya estaremos en condiciones de ejecutar los nuevos comandos.

Si deseamos utilizar los nuevos comandos RSX junto con los proporcionados con la rutina generadora de caracteres, deberemos cargar en memoria la presente rutina junto la que aparecía en un mumero anterior de esta misma revista, e inicializar los dos bloques de comandos RSX.

Esto se podrá conseguir confeccionando un programa Basic como el que se muestra a continuación:

```
10 MEMORY &8FFF
20 LOAD "CARAC1", &9500
30 LOAD "CARAC2", &9000
40 CALL &9500
50 CALL &9000
```

De esta forma serán utilizables tanto los nuevos comandos como los anteriores.

LISTADO ASSEMBLER

```
  10          ORG  #9000
  20          LD   DE,32
  30          LD   HL,CARAC
  40          CALL #BBAB
  50          LD   BC,TABLA
  60          LD   HL,ESPACE
  70          JP   #BCD1
  80 TABLA:   DEFW NAME1
  90          JP   UNO
 100          JP   DOS
 110          JP   TRES
 120 NAME1:   DEFM "T"
 130          DEFB "1"+#80
 140          DEFM "T"
 150          DEFB "2"+#80
 160          DEFM "T"
 170          DEFB "3"+#80
 180          DEFB 0
 190 ESPACE:  DEFS 4
 200 VALFA:   LD   (TIPO),A
 210          LD   L,(IX+2)
 220          LD   H,(IX+4)
 230          LD   (POSIC),HL
 240          LD   L,(IX+0)
 250          LD   H,(IX+1)
 260          LD   A,(HL)
 270          CP   80
 280          JP   NC,ERROR
 290          INC  HL
 300          LD   E,(HL)
 310          INC  HL
 320          LD   D,(HL)
 330          EX   DE,HL
 340          LD   DE,NAME
 350          LD   B,A
 360 MOS:     LD   A,(HL)
 370          LD   (DE),A
 380          INC  HL
 390          INC  DE
 400          DJNZ MOS
 410          LD   A,255
 420          LD   (DE),A
 430          JP   INIC
 440 UNO:     CP   3
 450          JP   NZ,ERROR
 460          LD   A,1
 470          JP   VALFA
 480 DOS:     CP   3
 490          JP   NZ,ERROR
 500          LD   A,2
 510          JP   VALFA
 520 TRES:    CP   3
 530          JP   NZ,ERROR
 540          LD   A,3
 550          JP   VALFA
 560 PONES:   LD   A,252
 570          LD   HL,MATRIZ
 580          CALL #BBA8
 590          LD   A,253
 600          LD   HL,MATRIZ+8
 610          CALL #BBA3
 620          LD   A,254
 630          LD   HL,MATRIZ+16
 640          CALL #BBA8
 650          LD   A,255
 660          LD   HL,MATRIZ+24
 670          CALL #BBA8
 680          RET
 690 INIC:    LD   HL,NAME
 700 BUC:     LD   A,(HL)
 710          CP   255
 720          RET  Z
 730          PUSH HL
 740          CALL #B8A5
 750          LD   A,(TIPO)
 760          CP   1
 770          PUSH AF
 780          CALL Z,TIPO1
 790          POP  AF
 800          CP   2
 810          PUSH AF
 820          CALL Z,TIPO2
 830          POP  AF
 840          CP   3
 850          CALL Z,TIPO3
 860          POP  HL
 870          INC  HL
 880          JR   BUC
 890
 900
 910 TIPO1:   LD   DE,MATRIZ
 920          LD   B,8
 930          CALL GRANDE
 940          CALL PONES
 950          CALL PINTA1
 960          RET
 970
 980 TIPO2:   CALL ANCHO
 990          CALL PONES
1000          CALL PINTA2
1010          RET
1020
1030 TIPO3:   CALL ANCHO
1040          LD   HL,MATRIZ
1050          LD   DE,MATRI1
1060          LD   BC,16
1070          LDIR
1080          LD   HL,MATRI1
1090          LD   DE,MATRIZ
1100          LD   B,8
1110          CALL GRANDE
1120          LD   HL,MATRI1+8
1130          LD   DE,MATRIZ+16
1140          LD   B,8
1150          CALL GRANDE
1160          CALL PONES
1170          CALL PINTA3
1180          RET
1190
1200
1210 GRANDE:  LD   A,(HL)
1220          LD   (DE),A
1230          INC  DE
1240          LD   (DE),A
1250          INC  DE
1260          INC  HL
1270          DJNZ GRANDE
1280          RET
1290
1300
1310 ANCHO:   LD   DE,MATRIZ
1320          LD   B,8
1330 BUCAN:   XOR  A
1340          PUSH DE
1350          BIT  7,(HL)
1360          CALL NZ,PON67
1370          BIT  6,(HL)
1380          CALL NZ,PON45
1390          BIT  5,(HL)
1400          CALL NZ,PON23
1410          BIT  4,(HL)
1420          CALL NZ,PON01
1430          LD   (DE),A
1440          PUSH HL
1450          EX   DE,HL
1460          LD   DE,8
1470          ADD  HL,DE
1480          EX   DE,HL
1490          POP  HL
1500          XOR  A
1510          BIT  3,(HL)
1520          CALL NZ,PON67
1530          BIT  2,(HL)
1540          CALL NZ,PON45
1550          BIT  1,(HL)
1560          CALL NZ,PON23
1570          BIT  0,(HL)
1580          CALL NZ,PON01
1590          LD   (DE),A
1600          POP  DE
1610          INC  DE
1620          INC  HL
1630          DJNZ BUCAN
1640          RET
1650
1660 PON67:   SET  6,A
1670          SET  7,A
1680          RET
1690 PON45:   SET  4,A
1700          SET  5,A
1710          RET
1720 PON23:   SET  2,A
1730          SET  3,A
1740          RET
1750 PON01:   SET  0,A
1760          SET  1,A
1770          RET
1780
1790
1800 PINTA2:  LD   HL,(POSIC)
1810          CALL #BB75
1820          LD   A,252
1830          CALL #BB5A
1840          LD   HL,(POSIC)
1850          INC  H
1860          CALL #BB75
1870          LD   A,253
```

```
1880            CALL #885A
1890            LD   HL,(POSIC)
1900            INC  H
1910            INC  H
1920            LD   (POSIC),HL
1930            RET
1940 PINTA1:    LD   HL,(POSIC)
1950            CALL #8875
1960            LD   A,252
1970            CALL #885A
1980            LD   HL,(POSIC)
1990            INC  L
2000            CALL #8875
2010            LD   A,253
2020            CALL #885A
2030            LD   HL,(POSIC)
2040            INC  H
2050            LD   (POSIC),HL
2060            RET
2070 PINTA3:    LD   HL,(POSIC)
2080            CALL #8875
2090            LD   A,252
2100            CALL #885A
2110            LD   HL,(POSIC)
2120            INC  H
2130            CALL #8875
2140            LD   A,254
2150            CALL #885A
2160            LD   HL,(POSIC)
2170            INC  L
2180            CALL #8875
2190            LD   A,253
2200            CALL #885A
2210            LD   HL,(POSIC)
2220            INC  H
2230            INC  L
2240            CALL #8875
2250            LD   A,255
2260            CALL #885A
2270            LD   HL,(POSIC)
2280            INC  H
2290            INC  H
2300            LD   (POSIC),HL
2310            RET
2320 ERROR:     LD   HL,TXTE
2330 BUCER:     LD   A,(HL)
2340            CP   255
2350            RET  Z
2360            CALL #885A
2370            INC  HL
2380            JR   BUCER
2390 TXTE:      DEFM "ERRDR EN PARAMETROS"
2400            DEFB 255
2410 POSIC:     DEFW #0505
2420 TIPO:      DEFS 1
2430 NAME:      DEFS 81
2440 MATRI2:    DEFS 32
2450 MATRI1:    DEFS 32
2460 CARAC:     EQU  #9FFB
```

TABLA DE ETIQUETAS

ANCHO	9106	BUC	9097	BUCAN	910B
BUCER	91D5	CARAC	9FF8	DOS	905F
ERROR	91D2	ESPACE	9024	GRANDE	90FD
INIC	9094	MATRI1	9267	MATRI2	9247
MOS	9049	NAME	91F6	NAME1	901D
PINTA1	917A	PINTA2	915A	PINTA3	9199
PON01	9155	PON23	9158	PON45	9148
PON67	9146	PONES	9073	POSIC	91F3
TABLA	9012	TIPO	91F5	TIPO1	9089
TIPO2	90C8	TIPO3	90D2	TRES	9069
TXTE	91DF	UNO	9055	VALFA	902B

CARGADOR BASIC

```
10 FOR N=&9000 TO &91F5
20 READ A:SUMA=SUMA+A
30 POKE N,A
40 NEXT
50 IF SUMA<>67140 THEN PRINT "ERROR
 EN DATAS":END
60 DATA 17,32,0,33,248,159,205
70 DATA 171,187,1,18,144,33,36
80 DATA 144,195,209,188,29,144,195
90 DATA 85,144,195,95,144,195,105
100 DATA 144,84,177,84,178,84,179
110 DATA 0,0,0,0,0,50,245
120 DATA 145,221,110,2,221,102,4
130 DATA 34,243,145,221,110,0,221
140 DATA 102,1,126,254,80,210,210
150 DATA 145,35,94,35,86,235,17
160 DATA 246,145,71,126,18,35,19
170 DATA 16,250,62,255,18,195,148
180 DATA 144,254,3,194,210,145,62
190 DATA 1,195,40,144,254,3,194
200 DATA 210,145,62,2,195,40,144
210 DATA 254,3,194,210,145,62,3
220 DATA 195,40,144,62,252,33,71
230 DATA 146,205,168,187,62,253,33
240 DATA 79,146,205,168,187,62,254
250 DATA 33,87,146,205,168,187,62
260 DATA 255,33,95,146,205,168,187
270 DATA 201,33,246,145,126,254,255
280 DATA 200,229,205,165,187,58,245
290 DATA 145,254,1,245,204,185,144
300 DATA 241,254,2,245,204,200,144
310 DATA 241,254,3,204,210,144,225
320 DATA 35,24,222,17,71,146,6
330 DATA 8,205,253,144,205,115,144
340 DATA 205,122,145,201,205,6,145
350 DATA 205,115,144,205,90,145,201
360 DATA 205,6,145,33,71,146,17
370 DATA 103,146,1,16,0,237,176
380 DATA 33,103,146,17,71,146,6
390 DATA 8,205,253,144,33,111,146
400 DATA 17,87,146,6,8,205,253
410 DATA 144,205,115,144,205,153,14
5
420 DATA 201,126,18,19,18,19,35
430 DATA 16,248,201,17,71,146,6
440 DATA 8,175,213,203,126,196,70
450 DATA 145,203,118,196,75,145,203
460 DATA 110,196,80,145,203,102,196
470 DATA 85,145,18,229,235,17,8
480 DATA 0,25,235,225,175,203,94
490 DATA 196,70,145,203,86,196,75
500 DATA 145,203,78,196,80,145,203
510 DATA 70,196,85,145,18,209,19
520 DATA 35,16,198,201,203,247,203
530 DATA 255,201,203,231,203,239,20
1
540 DATA 203,215,203,223,201,203,19
9
550 DATA 203,207,201,42,243,145,205
560 DATA 117,187,62,252,205,90,187
570 DATA 42,243,145,36,205,117,187
580 DATA 62,253,205,90,187,42,243
590 DATA 145,36,36,34,243,145,201
600 DATA 42,243,145,205,117,187,62
610 DATA 252,205,90,187,42,243,145
620 DATA 44,205,117,187,62,253,205
630 DATA 90,187,42,243,145,36,34
640 DATA 243,145,201,42,243,145,205
650 DATA 117,187,62,252,205,90,187
660 DATA 42,243,145,36,205,117,187
670 DATA 62,254,205,90,187,42,243
680 DATA 145,44,205,117,187,62,253
690 DATA 205,90,187,42,243,145,36
700 DATA 44,205,117,187,62,255,205
710 DATA 90,187,42,243,145,36,36
720 DATA 34,243,145,201,33,223,145
730 DATA 126,254,255,200,205,90,187
740 DATA 35,24,246,69,82,82,79
750 DATA 82,32,69,78,32,80,65
760 DATA 82,65,77,69,84,82,79
770 DATA 83,255,5,5,0,0,0
```

Tu cuarta pieza y tu cuarto número

Recorta y pega esta pieza en su lugar.

4

Recorta este número y guárdalo hasta que tengas los restantes, después pégalo en su lugar correspondiente, de forma que las sumas horizontales y verticales coincidan (15).

IMPRESION SIN PROBLEMAS

La impresora incluida con el PCW 8256/512 se diferencia de las demás en la casi ausencia de «switches» y controles externos, ya que estas funciones se realizan en todo momento desde el software suministrado con el ordenador, bien sea el procesador de textos Locoscript, o el Mallard Basic.

Todas las facilidades de la impresora del PCW se pueden manipular directamente desde Locoscript y todos los comandos de Locoscript son compatibles con la impresora.

Si usted nunca ha tenido que instalar una particular impresora para un determinado ordenador, no puede hacerse una idea exacta de la gran ventaja, en términos de tiempo y nervios, que representa el estrecho lazo que hay entre la impresora del PCW y el ordenador: letras itálicas, subrayadas, negritas, todas se imprimen sin fallo alguno y rápidamente con la sola elección del ítem del menú correspondiente.

La tecla IMPR

Se le comunica al PCW que deseamos que tome el control de la impresora mediante la tecla IMPR, situada a la izquierda de la tecla SAL en la fiola inferior del teclado. Esta tecla causa que Locoscript asuma el modo impresora *(«Printer Mode»)* y liste los comandos de control de impresora en la barra de **«status»**.

Otra manera de hacer lo mismo es jugar con los escasos controles que SI están físicamente en la impresora, como el **«paper-feed»** o la barra de sujeción del papel *(basta levantarla y volverla a bajar)*.

Suponiendo que ninguno de los menús-persiana *(«pull dawn menus»)* esté activo, se puede volver en cualquier momento del modo impresora al modo edición de texto o al manejo de ficheros del disco pulsando la tecla EXIT. Si uno quiere abortar en medio de una operación de cualquier menú, se puede usar la tecla CAN en la forma acostumbrada.

Un poco de claridad

No obstante todas estas facilidades, debemos admitir que el uso de las varias teclas de función disponibles, en modo control de impresora, no son tan claras como deberían ser. Lo mismo, por desgracia, se puede decir de los diferentes nombres de teclas que se pueden ver en la barra de **«status»** *(la barra de «cómo está el patio», vox populi)*.

A causa de esto, es deseable establecer un procedimiento fijo que se pueda usar para preparar la impresora al comienzo de cada sesión de impresión. Siguiéndolo con cuidado, nos aseguramos de que nuestro precioso documento, SIEMPRE, presente el aspecto que realmente queremos que tenga.

Comenzamos ajustando el único control, propiamente dicho, que está en la misma impresora: le podríamos llamar **«la palanca de fuerza de impresión»** *(figura 1)*. Se encuentra en el interior, a la derecha, y, para asegurarse de que exista una máxima claridad en lo impreso, debiera estar colocada a baja presión *(hacia atrás)* para hojas sueltas, y a alta presión *(hacia adelante)*, para múltiples copias. El truco está en seleccionar la menor presión consistente con la más alta calidad *(experimentación al canto)*. La siguiente tarea es adaptar la impresora al tamaño de papel que usaremos y a la calidad de letra requerida.

Esto se consigue pulsando f1, que muestra en pantalla un menú que permite al usuario elegir entre baja

y alta calidad, y entre papel continuo y hojas sueltas *(figura II)*.

Se pueden seleccionar las opciones que sean como de costumbre: con la barra espaciadora y la tecla «+».

Normalmente, no está nada mal seleccionar seis líneas por pulgada, aunque ocho, como describiremos luego, también pueden valer.

La longitud prevista por defecto para hojas sueltas corresponden al papel tipo A4, mientras que para papel continuo se ha previsto el de computadora típico, el que le van a vender a usted en la tienda.

Una vez que el papel ha sido «cargado» en la impresora mediante el control de carga, puede ser necesario, todavía, ordenarla unas cuantas cosas más, mediante la tecla de función f3.

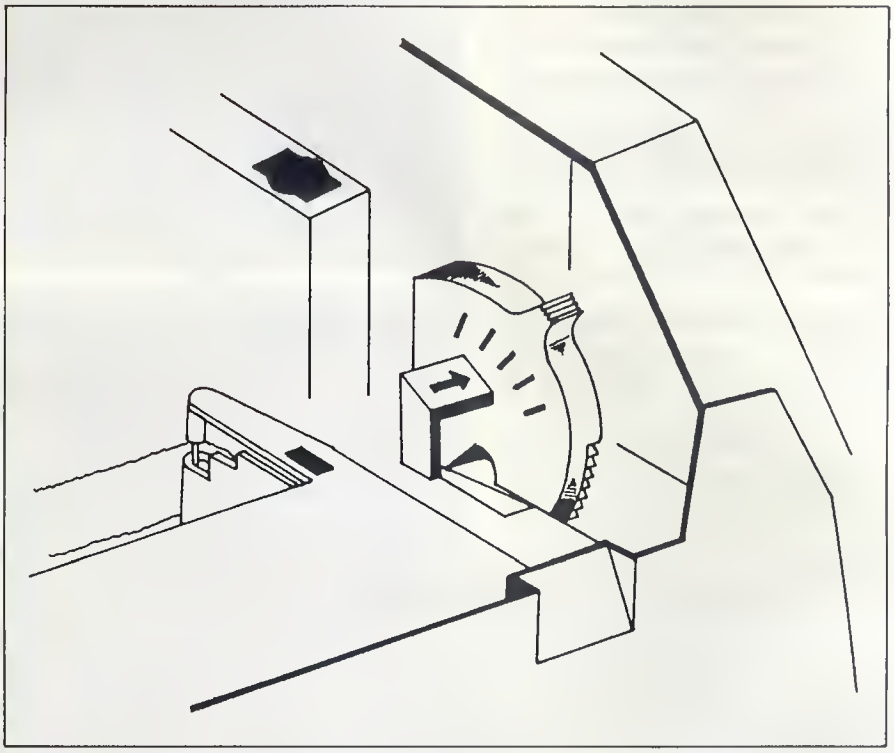
Este menú *(figura III)* ofrece dos acciones externas y una interna.

Menús de comandos

La primera acción externa obliga a la impresora a avanzar el papel una línea por página. Coloque el cursor sobre la opción adecuada y pulse «+». Es posible avanzar más líneas pulsando repetidamente esta tecla, pero, cuidado: no tiene autorrepetición.

La segunda acción externa es mover el cabezal de impresión del margen izquierdo y avanzarlo un cierto número de caracteres. Ponga la barra del cursor sobre la última línea del menú e introduzca el nuevo valor que desee.

La acción interna resetea el tope de página, y puede ser necesaria cuando hemos avanzado manualmente el papel, para así informar al ordenador de que la actual posición del cabezal de impresión es el nuevo tope de página.



En todos los casos anteriores, pulsando ENTER confirmamos nuestra elección y salimos del menú.

Lo descrito hasta ahora, en cuanto a controles de impresora, son los que necesitará en la inmensa mayoría de los casos excepto, quizás, cuando altere la calidad de letra.

Después de una serie de pruebas, si se encuentra contento con los resultados, pulse SAL para abandonar el modo de control de impresora.

Subsecuentes impresiones de sus documentos, al dejar con SAL una sesión, de edición, o con P desde la pantalla del **«disc manager»** *(impresión de ficheros desde disco)*, presentarán las características definidas tal y como hemos indicado, aunque usted puede revisarlas y alterarlas en cualquier momento reentrando el modo de control de impresora mediante la tecla !MPR. El resto de las teclas en la línea de **«status»** de la impresora se usan, en general, para tomar alguna acción mientras está trabajando, o para corregir algún error cuando ha ocurrido.

En cada caso, habrá que volver a introducirse en el modo de control de impresora para dar las órdenes adecuadas. La impresora, mientras tanto, parará al final de la línea en curso o al comienzo de la siguiente.

Control de errores

Si algo va muy mal mientras se está imprimiendo, el papel que se atasca u otra cosa igualmente catastrófica, la tecla IMPR detendrá la impresora casi inmediatamente. Comenzará a imprimir de nuevo a una suave pulsación de SAL. Para prevenir, incluso esto, no queda más que poner la impresora **«off line»**, volverla a encender y aquí no ha pasado nada.

La función que realiza este milagro es la 8, y se dispara mediante f8, que actúa como un interruptor si/no, encendido/apagado.

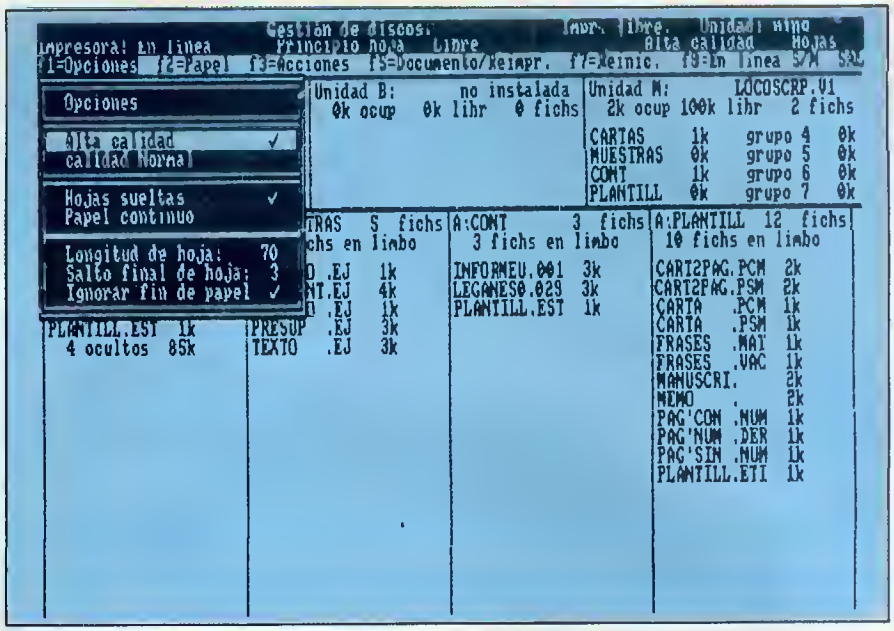
Si, por lo que sea, tiene que colocarse «off-line», debe restaurar, más tarde, dos cosas: el estado «on-line» (f8) y la posibilidad de que la impresora reconozca las señales del ordenador SAL.

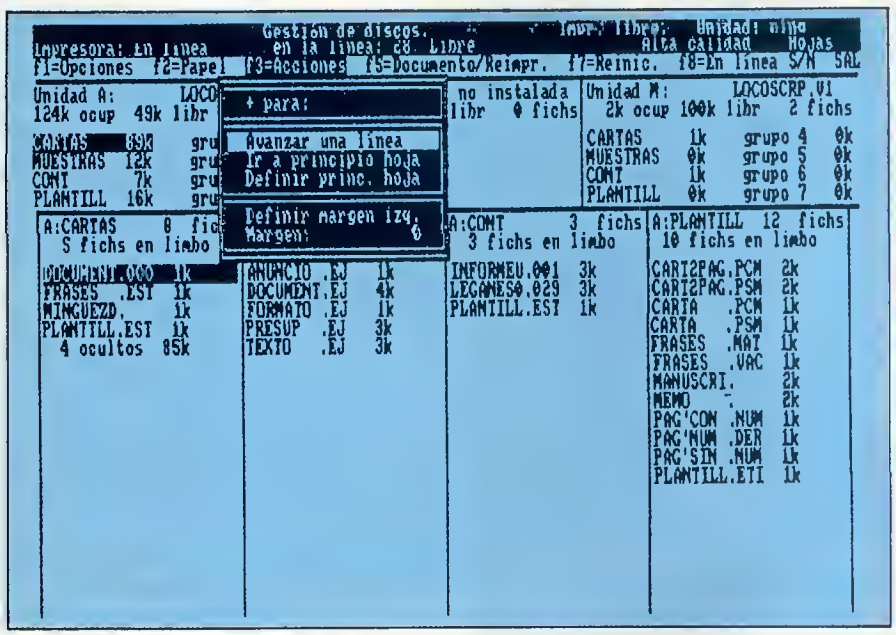
Ocasionalmente, se necesitará detener la impresión en medio de un documento, quizás porque se dé cuenta de la existencia de un error. Esto no se puede hacer solamente con la tecla PTR; hay que recurrir a f7 *(RESET)*, que también sirve para resetear la impresora y llevar la cabeza de impresión a su posición de «descanso».

El PCW tiene una característica muy avanzada, algo semejante, de cara al usuario, a la multitarea: se puede imprimir un documento mientras otro se está editando en pantalla *(por **«editar»** entendemos corregir y/o crear)*. El menú que controla esto se obtiene mediante f5, que da detalles del documento que se está imprimiendo o nos informa que la impresora no está en uso.

Un segundo propósito del menú «f5» es permitir la reimpresión de todo o parte del documento en curso, en el caso de que un error de cualquier tipo hubiera ocurrido en el interin.

Sólo hay otra ocasión en la que se necesite retornar al modo de control de impresora, y es cuando la impresora se detiene y la línea de **«status»** muestra la fase **«Esperando por papel»**, aunque dicho papel se encuentre, de hecho, cargado.





Esto puede suceder si está usando papel continuo, pero no ha inicializado los valores del menú controlado por f1, de acuerdo a ello, o si se ha introducido el papel **«a mano»**, en lugar de automáticamente. Pulsando f2 y ENTER la impresora continuará normalmente.

Al usar hojas sueltas, la impresora se detiene al final de cada página para permitir la inserción de una hoja nueva, como cabría esperar. Al **«aparecer»** una nueva hoja, automáticamente se vuelve al modo de control de impresora, así que, cuando esté listo, pulse SAL para comenzar la impresión en la nueva página.

Finalmente, una imagen *(hard copy)* de la pantalla aparecerá en la impresora en cualquier momento si se pulsa a la vez EXTRA y IMPR. Siempre será en alta calidad, y con un cierto aire de **«negativo»**, es decir, áreas llenas de verde aparecerán en negro. Muchos fabricantes de impresoras sostienen que un uso exagerado de esta facilidad dañará el cabezal de impresión; nosotros no sabemos nada de tales problemas con el PCW, pero un poco de precaución nunca sobra: uso, pero no abuso.

La mayoría de los comandos de Locoscript se pueden dar de diferentes maneras. Como es lógico, la manera más rápida es la que involucra menos pulsaciones de teclas. No obstante, es la más propensa a error hasta que se adquiere un dominio más que aceptable de la multitud de órdenes del procesador de textos del PCW. En principio, el uso de menús es muy aconsejable, y no retrasará demasiado su trabajo.

Paso Cualquier programa, incluso los protegidos, de cinta a disco o de disco a cinta. Total seguridad de carga. Tel. (91) 445 10 02. Preguntar por Juan, tardes.

Se hacen programas de gestión a medida para usuarios de **Amstrad** PCW-8256: facturación, clientes, proveedores, almacén, etc. y aplicaciones enteras. Preferiblemente en la provincia de Asturias. Interesados llamar al Tel. (985)46 73 32. Preguntar por Angel.

Vendo ZX Spectrum Plus, con su correspondiente transformador y cables para electricidad, TV y cassette, más dos manuales de referencia (uno en castellano) todo ello por 25.000 pts. Llamar al (976) 56 53 98 o escribir a Jesús Ruiz Vela. C/ Pedro Cuarta 11 12-D 50009 (Zaragoza).

Vendo los libros siguientes totalmente nuevos: «Música y sonidos con **Amstrad**», de Jeremy Vine, por 800 pts. (en la calle cuesta 1.200). «Juegos sensacionales para **Amstrad**», de J. Gregory, por 1.600 pts. También cambio los número 7 y 32 de Microhobby **Amstrad** por cualquiera de los siguientes: 6, 10, 13...20. Carlos F. Cano Lanzadera. C/ A. Sáinz de Baranda, 99. Tel. 273 00 18. 28007 Madrid.

Desearía intercambiar juegos **Amstrad** con usuarios de Barcelona o provincia. Me gustaría conseguir Capián a cambio de juegos. Ponerse en contacto con, José Francisco Izquierdo Marena. Avda. San Esteban, 59, 4.°. O si lo desean, llamar al Tel. (91) 870 10 71.

CONVERTIDOR TV PARA AMSTRAD

Conservando la misma línea sobria marcada por sus últimos productos, MHT vuelve a sorprendernos con un nuevo periférico que consigue un nuevo aprovechamiento de nuestro ordenador como TELEVISION.

Desde que el Home-Computer fue tal, todas las marcas aprovecharon los televisores domésticos como saludable medida para la reducción de costos en el equipo. La llegada de Alan Sugar al terreno de la microinformática cambió totalmente las limitaciones establecidas hasta la fecha, incluyendo un monitor en el equipo a un precio altamente competitivo.

Ahora **MHT** ha arremetido con acción inversa: convertir el monitor del ordenador en un televisor. A fin de cuentas, si hemos hecho una inversión en un monitor, ¿por qué no sacarle todo el partido?

El equipo consiste básicamente en una caja negra, perfectamente diseñada para ser colocada debajo del monitor, con el consiguiente ahorro de espacio. Las únicas precauciones y mantenimiento necesario es no exponer el aparato al calor ni la humedad y desenchufarlo cuando no vaya a ser utilizado durante algunos días.

Conexión del convertidor C-10

La conexión del equipo es muy sencilla y se realiza en menos de medio minuto. Para realizarla basta con enchufar el conector de cinco patas que sale del monitor al aparato, la antena a la toma trasera incorporada en el equipo y el enchufe del C-10 a la red.

Una vez realizadas las conexiones y activada la corriente habrá que sintonizar el convertidor. El número de presintonías que dispone es de 10, lo cual es más que suficiente si consideramos que a lo sumo necesitaremos cuatro: una para **TV-1,** otra para **TV-2,** otra para el canal **autonómico** y una última para el vídeo, siempre y cuando, naturalmente, dispongamos de éste.

Para sintonizar estos canales basta con girar los diales correspondientes, tal y como se realiza con cualquier televisor doméstico. Es obvio decir que, en el caso de disponer de vídeo, no conectaremos la antena al **C-10**, sino que conectaremos ésta al vídeo y la salida del vídeo directamente al convertidor. Por el diseño electrónico interior es más que recomendable utilizar el canal 8 para la reproducción de la señal proveniente de un vídeo.

Dado que el monitor del **Amstrad** no incorpora altavoz, el equipo dispone de uno interior, con una potencia sonora muy alta tanto en volumen como en calidad.

Para los más técnicos, el convertidor trabaja bajo las normas del CCIR y sistema de color PAL disponiendo de tres bandas de frecuencia, I, III y UHF. En la salida de un vídeo se dispone de una señal normalizada tipo vídeo compuesto con una amplitud de 1 V pp. Esta señal se puede utilizar tanto para reproducir como para grabar, para lo que bastará con cambiar el conmutador de modo, situado en la parte trasera, a la posición de salida.

El conector tipo **JACK** que sale del monitor no es necesario conectarlo en ningún sitio, así que nadie se pregunte dónde demonios ponerlo y mucho menos conectarlo a la toma

Banco de PRUEBAS

dose una calidad y un volumen muy superior. Estos mismo conectores son indispensables cuando queramos realizar una grabación en vídeo, a no ser que prescindamos del sonido y queramos grabar una película muda.

En resumen, la iniciativa de MHT de comerciolizar este, me atrevo a llamar, periférico, constituye una brillante idea ya que por un precio casi rídiculo, unas 20.000 pesetas, tenemos la posibilidad de disfrutar de un televisor completo en color, o en fósforo verde claro.

CARACTERISTICAS TECNICAS

- *Fabricante: MHT Ingenieros*
- *Distribuidor: LSB, S. A.*
- *Fecha de lanzamiento: 22/09/86*
- *Salida RGB-Lineal*
- *Entrada y salida de vídeo*
- *Entrada y salida de audio*
- *Amplificador de sonido y altavoz incorporado*
- *PVP (aprox.): 22.000 + IVA*

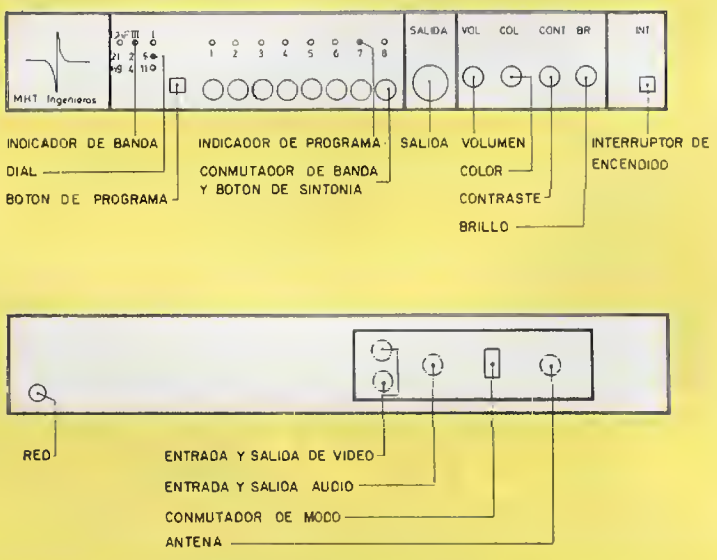
lateral de 12 voltios que tiene el propio monitor.

La información visual que nos proporciona el aparato se basa en LEDS; dispone de seis indicadores en color rojo que informan sobre la situación del equipo, ON/OFF, banda elegida, canal seleccionado... El indicador de dial le da una orientación de la zona de banda que está sintonizando; si estuviese sintonizando la banda I y el piloto, que indica este dial, se encontrase más encendido significaría que se encuentra más cerca del canal 2. Y así para cada una de las restantes sintonías.

A parte de estos controles más o menos sofisticados, el equipo dispone de los clásicos que pueden encontrarse en un televisor: brillo, contraste, volumen y color.

En la parte trasera, junto a la toma de antena, se encuentran dos conexiones para la entrada y salida de audio, lo que nos permite conectar el equipo a un amplificador lográn-

SITUACION Y FUNCION DE LOS CONTROLES



GLADIATOR

¡lucha por tu libertad!

Elige entre 45 armas diferentes

SPECTRUM
COMMODORE
AMSTRAD DISK
AMSTRAD

Si están agotados en tu tienda habitual ¡¡LLAMANOS!!

Si deseas información y participar en los importantes sorteos que ZAFICHIP celebrará durante el año... ¡ESCRIBENOS!

ZAFICHIP

ZAFIRO SOFTWARE DIVISION
Paseo de la Castellana, 141. 28046 Madrid
Tel. 459 30 04. Tel. Barna. 209 33 65. Télex: 22690 ZAFIR E

CONDUCE TU AMSTRAD 464 A 512 K

... Y AGUANTA EL VERTIGO

VORTEX SP-512 – Prodigiosa expansión de memoria.

Esta es la Placa VORTEX SP-512, capaz por sí sola de transformar tu ya conocido y dominado AMSTRAD 464 en un prodigio: 512 K de memoria, que multiplican hasta el vértigo las posibilidades de tu ordenador, gracias al sistema operativo VORTEX V-DOS incluido en la ROM.

Acude con tu AMSTRAD 464 a uno de los Servicios especializados VORTEX.

Una sencilla operación y la Placa VORTEX queda instalada. Se ha realizado la transformación que te lleva a una nueva frontera. Pasa a "RAM EXPANSION". ¿Estás preparado?

AMSTRAD es marca registrada de Amstrad PACKECOMP

vortex
COMPUTERSYSTEME

La Placa VORTEX SP-512 te abre un nuevo mundo de posibilidades de uso de tu ordenador.

Su instalación en el AMSTRAD 464 se realiza en los Servicios especializados VORTEX, quedando bajo la garantía PROTOMEC.

Ahora, carga a tu Amstrad hasta 2 Megas...

Con los Sistemas Operativos y las Unidades de Disco VORTEX V-DOS estás en condiciones, no sólo de preparar y desarrollar programas más largos y complejos, sino de utilizar, de verdad, hasta 2 MEGAS de datos.

El sistema operativo VORTEX V-DOS te lo permite. Es tan potente que, por ejemplo, puedes acceder a 16 ficheros directos, además de dos secuenciales.

Ficheros, Datos y Programas, Bases de Datos... Para dar servicio a esta capacidad de almacenamiento y de ampliación operativa, están los Sistemas Operativos y las Unidades de Disco VORTEX V-DOS, listos para ser conectados a los AMSTRAD 464, 664 y 6128.

Acude a un Servicio VORTEX y descubre la nueva frontera de tu AMSTRAD.

Infórmate en VORTEX. Tel. (91) 675 75 99
Avda. de la Constitución, 260.
Torrejón de Ardoz. MADRID.
O en cualquiera de los Servicios especializados VORTEX.

DE ESPECIAL INTERES PARA AULAS DE INFORMATICA DE ACADEMIAS Y COLEGIOS.

SERVICIOS VORTEX

ALAVA
DATAVI
Avda. Gasteiz, 29
Vitoria

ALICANTE
AUDIO- COLOR
Avda. Maisonnave, 17

AVILA
COMERCIAL ROCHA, S. A.
C/ Arévalo, 2

BADAJOZ
DONCEL
C/ Hernán Cortes, 3

BALEARES
DISTELEC
C/ Angel Guimerá, 23
Palma de Mallorca

BARCELONA
SVI DELEGACION CATALUÑA
Avda. Pau Claris, 165, Piso 3.º

BURGOS
E.I.S.A.
C/ Madrid, 4

CADIZ
MECANOGRAFIA GADITANA
C/ Rosario, 2

GRANADA
TECNICAS INFORMATICAS APLICADAS
Plaza Santo Cristo, 3 y 5

GUIPUZCOA
SOFT
C/ Cuesta de Aldapeta, s/n.
San Sebastián

DONOSTI COMPUCARD
Avda. de la Zurriola, 22 bis
San Sebastián

JAEN
OFIMATICA JAEN
Pasaje Maza, 7

LA CORUÑA
LOGICAR
Urbanización Galicia, 123
Sumoedo - Sada

LOGROÑO
EGUIZABAL C/ Doctores
Castro Viejo, 34

MADRID
MICRO WORLD
C/ Fernández de la Hoz, 46

MALAGA
MANIN INFORMATICA
Paseo Marítimo "Ciudad de Melilla", 25
Piso 11, Departamento C

NAVARRA
MICRO - HOBBY
C/ Arrieta, 40
Pamplona

PALENCIA
LA ESFERA
C/ Mayor Principal, 87

SALAMANCA
QUINTA AVENIDA
C/ España, 79

SANTA CRUZ DE TENERIFE
EQUINTESA
C/ San Sebastián, 74

SANTANDER
INSUMES
Centro Comercial
C/ Juan Ceballos, 9, bajo
Torrelavega

SEVILLA
MICROTIENDA I
C/ Acetuno, 8

TOLEDO
HOBBYLANDIA
C/ Fuenlabrada, s/n.
Polígono Industrial

VALLADOLID
CHIPS & TIPS
Plaza de Tenerías, 11

VIZCAYA
CHIP & TIPS
Alameda de Urquijo, 63
Bilbao

ZAMORA
VIDEO IMAGEN 21
C/ San Torcuato, 21

ZARAGOZA
DAJOL
C/ Cánovas, 31

vortex
COMPUTERSYSTEME